IMPLEMENTASI PROTOKOL UDP PERVASIVE MULTI DEVICE UNTUK PERANGKAT MYRIO BERBASIS LABVIEW (STUDI KASUS : SECURITY BOX)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Octavian Metta Wisnu Wardhana
NIM: 145150300111060

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI PROTOKOL UDP PERVASIVE MULTI DEVICE UNTUK PERANGKAT MYRIO BERBASIS LABVIEW (STUDI KASUS : SECURITY BOX)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun Oleh :
Octavian Metta Wisnu Wardhana
NIM: 145150300111060

Skripsi ini telah diuji dan dinyatakan lulus pada
Senin, 30 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Rizal Maulana , S.T., M.T., M.Sc.	Wijaya Kurniawan, S.T., M.T.
NIK: 2016078910091001	NIP: 19820125 201504 1 002

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 30 Juli 2018

Octavian Metta Wisnu Wardhana

NIM: 145150300111060

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Implementasi Sistem Pengenalan Perangkat dan Layanan Sensor dan Aktuator pada Rumah Cerdas Berbasis Arsitektur *Publish-Subscribe*" ini dapat terselesaikan. Laporan skripsi ini disusun dalam rangka memenuhi persyaratan untuk memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer

Penulis menyadari bahwa penyusunan laporan skripsi ini tidak lepas dari bantuan berbagai pihak baik secara langsung maupun tidak langsung. Dalam kesempatan ini penulis ingin menyampaikan ucapan terima kasih atas bantuannya kepada semua pihak, sehingga penulis dapat menyelesaikan laporan ini dengan baik. Ucapan terima kasih tersebut khususnya kepada :

1. Allah yang Maha Esa yang selalu memberikan petunjuk dan hikmah dalam penulisan ini.
2. Orang Tua dan Keluarga atas nasehat, kasih sayang serta dukungan materiil dan moril.
3. Bapak Rizal Maulana, S.T., M.T., M.Sc. dan Bapak Wijaya Kurniawan, S.T., M.T. selaku dosen pembimbing yang telah memberikan banyak dukungan dalam proses penyusunan skripsi ini baik teknis maupun non teknis.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku ketua jurusan Teknik Informatika.
5. Teman-teman The Sempols dan Khurinika atas dukungan, motivasi, serta kesediaan untuk menjadi teman diskusi selama pengerjaan skripsi.
6. Teman-teman semasa SMA (Dicho, Adiba, Nafa, Ratih, Farida) yang telah ikut membantu dan memberikan dukungan melancarkan proses pengerjaan skripsi.
7. Dan orang-orang yang selalu mendukung serta mendoakan kelancaran proses skripsi ini yang tidak bisa disebutkan satu per satu atas semua doa dan dukungannya.

Dengan segala keterbatasan pengetahuan yang dimiliki, penulis sadar bahwa penulisan laporan skripsi ini masih jauh dari kesempurnaan. Sehingga penulis berharap agar laporan skripsi ini dapat bermanfaat dan merupakan salah satu informasi yang berguna bagi pembaca.

Malang, 17 Juli 2018

Penulis

Octavian Metta Wisnu Wardhana

ABSTRAK

Pervasive computing merupakan salah satu aplikasi dari *ubiquotous computing* yang berguna untuk memudahkan pengoperasian komputer dengan meminimalkan kebutuhan akan interaksi dari manusia kepada komputer. Sedangkan UDP *Pervasive Multi Device* adalah protokol UDP yang dikembangkan menggunakan *pervasive computing* untuk dapat mengenali identitas perangkat lain tanpa diperlukan pengaturan secara manual dari pengguna. Pada penelitian ini protokol UDP *Pervasive* dapat diimplementasikan pada perangkat PC dan NI myRIO yang menggunakan program berbasiskan LabVIEW. Protokol UDP digunakan karena tidak membutuhkan proses *handshaking* sehingga mengurangi *delay* serta ukuran data yang lebih sedikit karena tidak adanya *acknowledgement field* dan *sequence field.* Setiap perangkat menggunakan desain sistem *state machine* ganda yang memiliki fungsi masing-masing mendeteksi perangkat lain secara otomatis dan sebagai aplikasi dari penggunaan alamat perangkat yang terdeteksi. Ketika pertama kali program dieksekusi, perangkat melakukan *broadcast* kepada semua perangkat. Kemudian perangkat melakukan *listening* untuk menerima data *broadcast* maupun balasan dari *broadcast.* Ketika menerima data *broadcast* atau balasan maka perangkat melakukan pengecekan duplikasi perangkat dan menyimpannya. Selanjutnya perangkat akan mengirimkan balasan ketika menerima data *broadcast* atau kembali *listening.* Dengan menggunakan *state machine* mendeteksi perangkat lain yang sama, maka perangkat dapat mengenali lebih dari satu perangkat lain pada jaringan yang sama. Hasil dari pengujian yang didapatkan semua pengujian *state based testing* berhasil 100%. Serta mendapatkan *discovery time* rata-rata 0,202754 detik untuk 1 myRIO dan 0,303201 detik untuk 2 myRIO. *Delay* pengiriman data dari PC kepada myRIO tidak lebih dari 2 detik.

Kata kunci: *pervasive computing, UDP Pervasive, multi device, LabVIEW, state machine*

ABSTRACT

Pervasive computing is one application of ubiquotous computing that is useful to facilitate the computer operation by minimizing the need for human interaction to the computer. While UDP Pervasive Multi Device is a UDP protocol developed using pervasive computing to be able to recognize the identity of other devices without the need for manual settings from the user. In this paper the UDP Pervasive protocol can be implemented on PC and NI myRIO devices using LabVIEW-based programs. UDP protocols are used because they do not require handshaking to reduce delay and smaller data sizes due to the absence of acknowledgment fields and sequence fields. Each device uses dual state machine system design that has function for detecting other devices automatically and as an application of using other devices address. When the program is first executed, the device broadcasts to all devices. Then the device does listening to receive broadcast data or replies from broadcast. When receiving broadcast data or replies then the device checks the duplication of the device and stores it.. The device then sends a reply when receiving broadcast or back to listening. By using the same state machine to detect all device, the device can recognize other devices more than one on the same network. The results of the test obtained that all state-based testing succeeded 100%. Discovery time averaging 0.202754 seconds for 1 myRIO and 0.303201 seconds for 2 myRIO. Sending data delay from PC to myRIO not more than 2 second.

Keywords : pervasive computing, UDP Pervasive, multi device, LabVIEW, state machine

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Salah satu perkembangan teknologi yang kentara adalah *Internet of Things* atau IoT. Perangkat seperti *smarthome, smartphone, smartTV,* adalah teknologi yang lahir tidak lepas peran IoT (Hashemi, Faghri, Rausch, & Campbell, 2016). *Internet of Things* adalah sistem yang terbentuk atas perangkat komputasi, mesin mekanik dan digital, objek, hewan, atau manusia yang memiliki identifikasi unik dan memiliki kemampuan untuk mengirimkan data melalui jaringan tanpa membutuhkan interaksi *human-to-human* atau *human-to-computer.* "*Thing*" dari IoT bisa berarti benda atau makhluk yang telah memiliki *IP Address* masing-masing (Wigmore, 2016). Dengan adanya *IP Address* maka tiap benda tersebut dapat terhubung dengan jaringan maupun internet sehingga "*Thing*" dalam *Internet of Things* dapat disebut sebagai *Internet Nodes.* Perusahaan analisis Gartner memperkirakan akan ada 26 miliar perangkat yang terkoneksi pada tahun 2020 (Lamey, 2018).

Besarnya potensi dari *Internet of Things* tentu harus diiringi dengan mudahnya pengoperasian perangkatnya. Hal tersebut perlu ditunjang dengan teknologi *pervasive computing*. *Pervasive computing* merupakan bidang penelitian dari *ubiquitous computing* yang memberikan paradigma revolusioner untuk *computing models* (Dourish & Bell, 2011). Dalam model *pervasive* memiliki konsep *"invisible",* dan *"integration"* yang berarti sistem membutuhkan minimal interaksi dari manusia dan dapat terintegrasi dengan perangkat di lingkungannya. Dengan *pervasive computing* setiap perangkat diharapkan dapat mengenali perangkat lain secara otomatis serta dapat bertukar data satu sama lain. (Saha & Mukherjee, 2003). Jadi *pervasive computing* cocok diimplementasikan untuk mengotomatisasi perangkat *smarthome* untuk memudahkan penggunaan dan minim interaksi dari manusia.

Dalam mengimplementasikan perangkat *smarthome* membutuhkan salah satu protokol dari *Transport Layer* dari OSI model. Terdapat protokol TCP (*Transmission Control Protocol*) dan protokol UDP (*User Datagram Protokol)* pada lapisan *Transport* (Kurose & Ross, 2010). Protokol UDP lebih cocok diaplikasikan pada perangkat *smarthome* karena tidak terhambat oleh proses penerimaan data dan tidak membutuhkan *sequence field* dan *acknowledgement field* sehingga ukuran data menjadi lebih kecil (Snoeren, 2000).

Dibutuhkan perangkat *embedded* yang dapat diimplementasikan oleh *pervasive computing* serta dapat terhubung jaringan dengan mudah. Perangkat *embedded* dari National Instruments yang bernama NI myRIO dinilai dapat menampung kebutuhan dari *pervasive computing* dari perangkat *smarthome* karena memiliki sensor accelerometer serta memiliki fitur untuk dapat terhubung dengan WiFi. Perangkat NI myRIO dapat diprogram dengan platform LabVIEW. LabVIEW dapat digunakan untuk program yang menggunakan desain sistem berbasiskan *state machine* dan komputasi paralel.

Berbagai penelitian yang menggunakan metode *pervasive computing* telah dilakukan. Penelitian yang berjudul "*Lightweight UDP Pervasive Protocol in Smart Home Environment Based on Labview"* merancangkan prototipe dari protokol UDP pervasif yang akan diimplementasikan menggunakan bahasa pemrograman LabVIEW (Kurniawan W. , Ichsan, Akbar, & Arwani, 2017). Desain sistem dari prototipe tersebut menggunakan arsitektur *client-host* sehingga terdapat perbedaan desain protokolnya. Penggunaan protokol UDP lebih cocok diterapkan dibandingkan protokol TCP sebab UDP lebih ringan dan dapat melakukan broadcast pada jaringan lokal.

Selain itu penelitian yang berjudul "*UDP Pervasive Protocol Implementation for Smart Home Environment on* MyRIO *using LabVIEW*" mengimplementasikan protoptipe dari protokol UDP pervasif dari penelitian sebelumnya ke perangkat *embedded* bernama myRIO. (Kurniawan, Ichsan, & Akbar, UDP Pervasive Protocol Implementation for Smart Home Environment on MyRIO using LabVIEW, 2017). Pada penelitian ini perangkat myRIO bertindak sebagai *client* dan PC (Personal Computer) bertindak sebagai *host*. Perangkat myRIO dapat digunakan sebagai perangkat *smarthome* karena membutuhkan daya yang kecil.

Berdasarkan penelitian sebelumnya yang telah dilakukan, protokol UDP *pervasive* dapat memudahkan penggunaan serta dapat diterapkan pada perangkat *embedded* lingkungan *smarthome* sehingga mendasari penulis untuk mengembangkan fitur dari desain sistem UDP *pervasive* dan mengimplementasikannya pada perangkat *embedded* myRIO. Pada penelitian ini penulis menambahkan fitur *multi*-perangkat sehingga setiap sebuah *device* dapat mengenali lebih dari satu perangkat yang lain pada sebuah jaringan. Dengan demikian diharapkan protokol UDP *pervasive* berhasil diuji dengan fitur multi perangkat.

1.2 Rumusan masalah

Dari permasalahan yang telah dibahas pada latar belakang, maka dapat dirumuskan beberapa permasalahan yang akan dibahas secara keseluruhan, di antaranya adalah:

1. Bagaimana rancangan desain sistem dari protokol UDP *pervasive* multi *device* untuk diterapkan pada perangkat myRIO?
2. Bagaimana mengimplementasikan hasil rancangan dari protokol UDP *pervasive* pada perangkat myRIO agar dapat mengenali perangkat lain pada jaringan yang sama secara otomatis?
3. Bagaimana hasil implementasi protokol UDP *pervasive* pada sistem *smarthome embedded* menggunakan perangkat myRIO?

1.3 Tujuan

1. Merancang desain sistem *multi devices* dari protokol UDP *pervasive* untuk diterapkan pada perangkat myRIO.

2. Mengimplementasikan rancangan protokol UDP *pervasive multi devices* agar perangkat myRIO dapat mengenali perangkat lain pada jaringan yang sama secara otomatis.
3. Dapat menguji implementasi protokol UDP *pervasive* pada sistem *smarthome embedded* menggunakan perangkat myRIO.

1.4 Manfaat

Dengan dirancangnya "*Implementasi Rancangan Protokol UDP Pervasive Multi Devices untuk Perangkat* myRIO *berbasis LabVIEW"* diharapkan dapat mengembangkan beberapa penelitian yang telah dilakukan sebelumnya, dapat diimplementasikan pada lingkungan *smarthome*, dan dapat dikembangkan lebih lanjut.

1.5 Batasan Masalah

Agar permasalahan yang telah dirumuskan lebih terfokus dan tidak terjadi pelebaran topik, maka penelitian ini dibatasi dalam hal:

1. Sistem hanya diimplementasikan pada perangkat *embedded* myRIO dan PC (*Personal Computer*) yang telah diprogram menggunakan bahasa pemrograman LabVIEW.
2. Pengujian hanya melingkupi *delay* komunikasi dan pengujian *state diagram.*
3. Pertukaran data menggunakan protokol UDP.
4. Perangkat yang diuji hanya berada dalam satu jaringan *Wireless Local Area Network* yang sama.
5. *Security Box* merupakan aplikasi dari penggunaan alamat perangkat lain dan digunakan sebagai studi kasus.

1.6 Sistematika penulisan

Sistematika penulisan dari penelitian ini direncanakan sebagai berikut:

BAB I : Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, manfaat, dan sistematika penulisan.

BAB II : Landasan Kepustakaan

Memuat kajian pustaka mengenai penelitian-penelitian yang pernah dilakukan dan menjelaskan dasar teori yang terkait dengan Implementasi Rancangan Alternatif Protokol UDP *Pervasive* untuk Perangkat myRIO berbasis LabVIEW.

BAB III : Metodologi

Memuat metode yang digunakan dalam penelitian. Terdiri atas studi literatur, analisis kebutuhan, perancangan dan implementasi, pengujian dan analisis, dan kesimpulan.

BAB IV : Rekayasa Kebutuhan Sistem

Membahas tentang analisis kebutuhan sistem dari kebutuhan perangkat lunak, kebutuhan perangkat keras, dan kebutuhan fungsional dengan rinci.

BAB V : Perancangan dan Implementasi

Membahas tentang rancangan protokol UDP *pervasive* yang akan diimplementasikan pada perangkat myRIO.

BAB VI : Pengujian

Membahas tentang pengujian dari hasil implementasi yang telah dilakukan pada tahap sebelumnya.

BAB VII : Penutup

Membahas tentang kesimpulan dari penelitian yang telah dilakukan pada semua tahap dan saran untuk keperluan pengembangan selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini membahas tentang tinjauan pustaka yang meliputi kajian pustaka dan dasar teori yang diperlukan. Kajian pustaka berisi perbedaan antara penelitian yang telah dilakukan. Dasar teori berisi teori yang diperlukan untuk menyusun penelitian yang diusulkan.

2.1 Kajian Pustaka

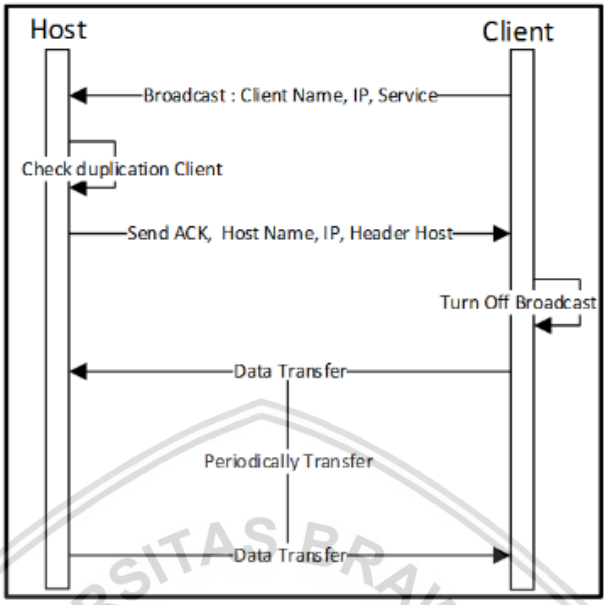
Kajian pustaka berisi beberapa jurnal atau *paper* yang dijadikan referensi oleh penulis untuk membantu penelitian. Kajian pustaka yang digunakan disebutkan pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka

No.	Judul	Penulis	Perbandingan	
			Kajian Pustaka	Penelitian Penulis
1	*Lightweight UDP Pervasive Protocol in Smart Home*	(Kurniawan W. , Ichsan, Akbar, & Arwani, 2017)	Menggunakan model *client-host* dan memiliki *state* yang berbeda tiap perangkat.	Menggunakan *state machine* mendeteksi perangkat lain yang sama tiap perangkat.
2	*UDP Pervasive Protocol Implementation for Smart Home Environment on* MyRIO *using LabVIEW*	(Kurniawan, Ichsan, & Akbar, UDP Pervasive Protocol Implementation for Smart Home Environment on MyRIO using LabVIEW, 2017)	Implementasi dari model *client-host*.	Implementasi dari *state machine* mendeteksi perangkat lain yang sama serta memiliki fitur *multi* perangkat.

Penelitian pertama yang berjudul "*Lightweight UDP Pervasive Protocol in Smart Home"* peneliti merancang desain protokol *pervasive* untuk domain *client* dan *host*. Kedua domain tersebut dioperasikan pada PC (*Personal Computer*). Desain sistem dari penelitian tersebut ditunjukkan pada Gambar 2.1. Berdasarkan Gambar 2.1, *host* berada dalam kondisi *listen*. Kemudian *client* mengirim *broadcast* yang berisi informasi *client* kepada *host* agar *host* dapat mengenali *client*.. *Host* akan melakukan pengecekan duplikasi apakah *client* yang baru mengirim *broadcast* telah terhubung dengan *host* sebelumnya atau tidak. Kemudian *host* mengirim pesan ACK dan informasi *host* sehingga *host* dikenali oleh *client*. Ketika *client* telah mengenal *host*, *client* akan menonaktifkan

broadcast tersebut. Transmisi data dapat dilakukan karena *host* dan *client* telah saling mengenal.



Gambar 2.1 Desain sistem protokol UDP *pervasive*

Sumber : **(Kurniawan W. , Ichsan, Akbar, & Arwani, 2017)**

Pada penelitian kedua yang berjudul "*UDP Pervasive Protocol Implementation for Smart Home Environment on* MyRIO *using LabVIEW*" telah mengimplementasikan protokol UDP *pervasive* dengan desain sistem seperti pada Gambar 2.1 pada perangkat NI myRIO dan diuji coba menggunakan skenario pengujian fungsional. PC (Personal Computer) bertindak sebagai *host*, sedangkan perangkat NI myRIO bertindak sebagai *client*.

Berdasarkan kedua penelitian di atas, penulis mengembangkan desain sistem alternatif yang memiliki *state machine* sama setiap perangkat sehingga dalam teori sebuah perangkat dapat mengenali lebih dari satu perangkat lain dalam jaringan yang sama.

2.2 UDP (User Datagram Protocol)

User Datagram Protocol adalah protokol yang dikembangkan oleh David P. Reed pada tahun 1980 dan didefinisikan pada RFC 768. UDP menggunakan model komuikasi *connectionless* yang berarti tidak membutuhkan proses *handshaking.* Pesan dari UDP juga dikirimkan sebagai datagram tanpa ada pesan ACK, tanpa mekanisme flow-control. (Kurose & Ross, 2010). Pada lingkungan *smart* home lebih cocok menggunakan UDP karena UDP adalah protokol yang ringan dan tidak terhambat oleh proses *handshaking* dan memorinya dilepas lebih awal. Serta dengan UDP dapat dilakukan proses *broadcasting* data ke semua perangkat yang terhubung pada *subnet* yang berperan pada *pervasive computing*.

2.3 Pervasive Computing

Pervasive computing adalah pengembangan dari perangkat *embedded* menjadi objek sehari-hari dan dapat beroperasi dengan efektif dengan interaksi minimal dari penggunanya. Perangkat *pervasive* selalu terhubung dengan jaringan. Tujuan dari *pervasive computing* adalah membuat perangkat menjadi "*smart*" serta dapat mengerti kondisi di lingkungannya (Shea & Tang, 2016). *Pervasive computing* memiliki berbagai tantangan sebagai ciri khasnya, yaitu :

1. *Scalability.*

   Lingkungan *pervasive computing* di masa depan akan mengalami pertumbuhan pengguna, aplikasi, perangkat, dan interaksinya dalam skala yang sangat besar. Dengan tumbuhnya kecerdasan lingkungan, maka banyaknya perangkat yang terhubung dengan lingkungan dan interaksinya dengan manusia juga ikut tumbuh.

   Pengembangan yang tradisional membutuhkan menciptakan aplikasi baru setiap terdapat perangkat baru sehingga dapat menimbulkan masalah skalabilitas aplikasi.

   Selanjutnya, aplikasi biasanya didistribusikan dan dipasang terpisah untuk setiap kelas perangkat dan prosesor. Dengan tumbuhnya jumlah perangkat, mendistribusikan dan memasang aplikasi untuk setiap perangkat menjadi tidak terkontrol, terutama pada area geografi yang luas.

2. *Heterogeneity.*

   Pervasive computing akan mengisi kekosongan pada suatu level untuk mengatasi masalah beragamnya perangkat ataupun protokol yang terdapat pada lingkungan.

   Saat ini, aplikasi biasa dikembangkan untuk perangkat dan sistem yang spesifik sehingga menimbulkan perbedaan versi dari aplikasi yang sama pada platform yang berbeda. Dengan bertambahnya heterogenitas, mengembangkan aplikasi yang dapat berjalan pada semua platform akan semakin sulit.

3. *Integration.*

   Dengan banyaknya perangkat dan aplikasi membuat proses integrasi *pervasive computing* pada lingkungan menjadi lebih kompleks. Sebagai contoh, server harus dapat mengatur ribuan klien yang terhubung bersamaan. *Pervasive Computing* membutuhkan koordinasi untuk dapat mengatur banyaknya perangkat atau aplikasi.

4. *Invisibility.*

   Pervasive Computing membutuhkan sedikit interaksi manusia kepada komputer sehingga komputer seolah-olah tak terlihat atau

invisible. Manusia dapat mengintervensi sistem cerdas apabila terdapat kegagalan otomatisasi sistem pervasif.

5. *Perception : Context awareness.*

   Pervasive Computing membutuhkan sistem dan perangkat yang dapat merasakan konteks. Sebagai contoh pada perangkat *mobile* dapat mengetahui pengguna sedang berkendara berdasarkan pergerakan pada lokasinya.

6. *Smartness : Contect management.*

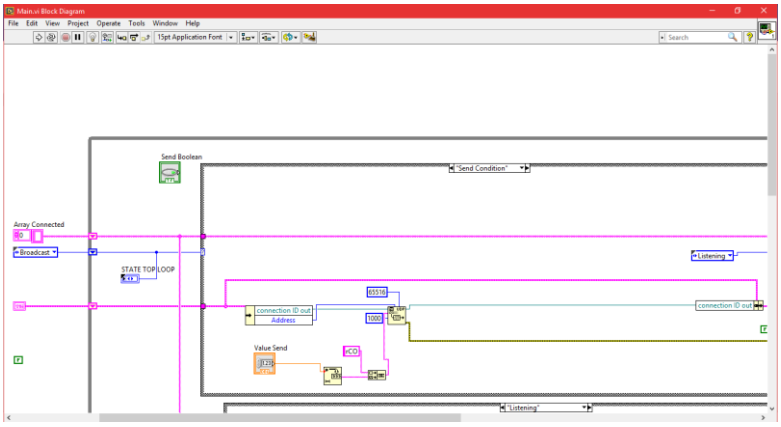
   Smartness membutuhkan *input sensing* yang akurat serta kontrol intelijen atau *output* antara mesin dan manusia. Sebagai contoh, sistem *pervasive computing* dapat mengatur suhu dan pencahayaan ruangan secara otomatis.

Dengan *pervasive computing* dapat memudahkan penggunaan dari perangkat *smart home* serta perangkat tersebut dapat mengenali lingkungannya.

2.4 LabVIEW

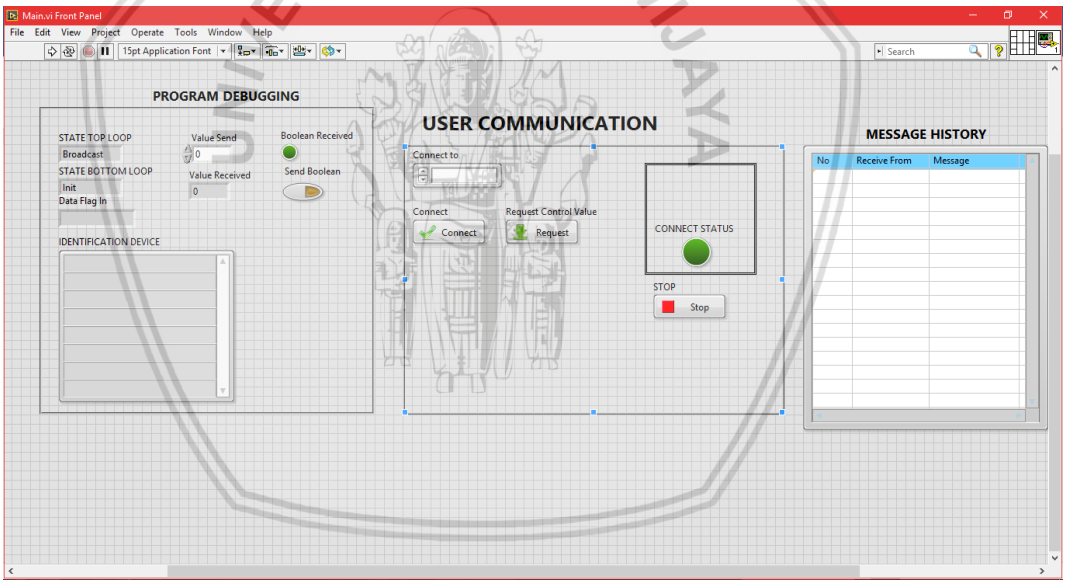
LabVIEW atau Laboratory Virtual Instrumentation Engineering Workbench adalah perangkat lunak yang digunakan sebagai platform desain sistem dan *development environment* untuk bahasa pemrograman visual dari National Instuments. Bahasa pemrograman visual yang digunakan bernama "G". Perangkat lunak ini biasa digunakan untuk pemrosesan data, akuisisi data, kontrol instrumentasi, dan virtualisasi data. (Halvorsen, 2016)

Platform ini menggunakan *dataflow programming* dari bahasa pemrograman "G". Alur eksekusinya ditentukan oleh struktur dari blok diagram yang terhubung dari beberapa *node* fungsi yang ada. *Node* akan segera dieksekusi setelah tersedia data *inputnya*. Pada LabVIEW juga mendukung pemrograman paralel dan dapat mengeksekusi beberapa *node* secara bersama oleh *scheduler* yang telah tersedia. Contoh dari penggalan program dari LabVIEW ditunjukkan pada Gambar 2.2.



Gambar 2.2 Contoh tampilan back panel.

LabVIEW juga terintegrasi dengan pembuatan *user interface* yang disebut Front panel seperti yang ditunjukkan oleh Gambar 2.3. Pada *Front Panel* dibuat oleh *control* dan *indicator. Indicator* digunakan sebagai *output,* sedangkan *control* digunakan sebagai *input*. Semua objek yang ada pada front panel akan nampak pada back panel sebagai terminal. Pada *back panel* berisi struktur dan *node* fungsi. Setiap *node* dihubungkan dengan *wire*.



Gambar 2.3 Contoh tampilan Front panel.

Platform LabVIEW dapat diimplementasikan desain sistem seperti *state machine, event-driven UI*, *Master slave*, *producer consumer*, atau *queued state machine with event-driven producer-consumer*.

2.5 NI myRIO

NI myRIO adalah perangkat *embedded* yang dibuat untuk pembelajaran mengenai pemrograman FPGA atau pemrograman *embedded*. NI myRIO menggunakan arsitektur LabVIEW RIO yang memiliki 4 komponen, yaitu

prosesor, FPGA, *input* dan *output*, serta perangkat lunak desain sistem (National Instruments, 2014). Program yang telah dibuat dari platform LabVIEW dapat diimplementasikan ke dalam perangkat NI myRIO.

Gambar 2.4 NI myRIO-1900

Sumber : **(National Instruments, 2014)**

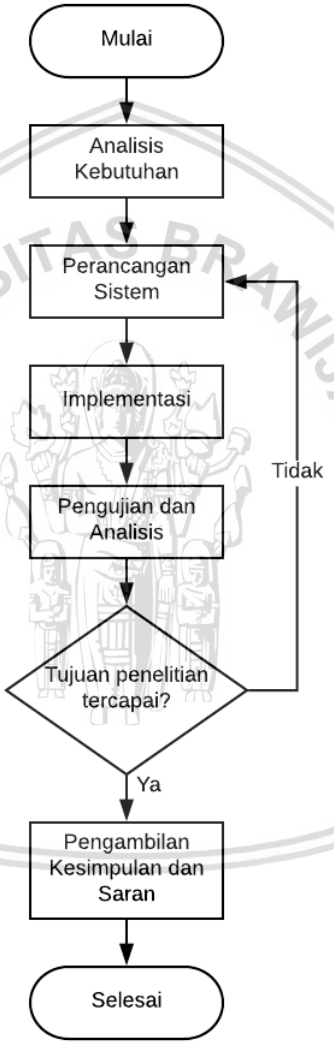
Berikut adalah spesifikasi dari NI myRIO-1900 (National Instruments, 2016):

- Prosesor FPGA Xilinx Z-7010 (667 MHz, 2 cores)
- 512 MB *Nonvolatile memory*.
- 256 MB DDR3 memory (533 MHz, 16 bits)
- *Wireless Radio* 802.11 b,g,n. 2.4 GHz. *Up to* 150 m.
- USB 2.0 *host* and device.
- *3 axis accelerometer*.
- 2 *ports* of 16 Digital I/O lines.
- Stereo audio *input* and *output*.

Perangkat NI myRIO digunakan pada penelitian ini karena mendukung pengimplementasi *parallel computing*, memiliki *Wireless Radio* dan sensor akselerometer, serta dapat diprogram menggunakan LabVIEW.

BAB 3 METODOLOGI

Dalam perancangan sistem terdapat proses-proses yang berhubungan agar penciptaan perancangan sistem dapat tersusun dengan baik. Bab ini menjelaskan tentang metode yang digunakan oleh peneliti beserta langkah-langkahnya dalam penelitian. Tahapan penelitian yang digunakan pada penelitian ini dapat digambarkan dalam bentuk diagram blok yang ditujukan pada Gambar 3.1



Gambar 3.1 Diagram Alir Metodologi Penelitian

3.1 Studi literatur

Tahapan studi literatur pada penelitian ini adalah mempelajari literatur dari beberapa sumber pustaka yang berkaitan dengan penelitian *"Implementasi*

Rancangan Alternatif Protokol UDP Pervasive untuk Perangkat myRIO *berbasis LabVIEW"* yang didapatkan dari buku, jurnal, dan situs terkait. Teori pustaka yang berkaitan dengan penelitian ini adalah:

1. *Pervasive Computing*

   Merupakan salah satu metode yang bertujuan untuk meminimalkan interaksi dari pengguna terhadap sebuah sistem untuk memudahkan penggunaan. Diterapkan pada penelitian ini untuk mengenali perangkat lain yang terdapat pada jaringan.

2. UDP *Protocol*

   Protokol UDP digunakan pada penelitian ini karena dinilai sebagai protokol yang ringan karena tidak memerlukan proses *handshaking*.

3. LabVIEW

   LabVIEW adalah bahasa pemrograman yang menerapkan *dataflow programming* dan mendukung komputasi paralel. Dapat diimplementasikan pada *Personal Computer* dan pada perangkat NI myRIO.

4. NI myRIO

   Perangkat yang mendukung komputasi paralel dan memiliki *wireless radio.* Diprogram menggunakan bahasa pemrograman LabVIEW.

3.2 Analisis Kebutuhan

Analisis kebutuhan sistem bertujuan untuk mendapatkan semua kebutuhan yang diperlukan sistem dalam penelitian ini yang akan dibangun dan diuji. Analisis kebutuhan dilakukan dengan mengidentifikasi kebutuhan sistem secara keseluruhan. Dalam kebutuhan sistem akan terjadi proses identifikasi perangkat yang digunakan baik secara *hardware* maupun *software*. Dengan adanya identifikasi ini maka nantinya dapat membantu serta mempermudah dalam desain sampai pembuatan sistem.

Berikut adalah penjelasan analisis kebutuhan dari penelitian *"Implementasi Rancangan Alternatif Protokol UDP Pervasive untuk Perangkat myRIO berbasis LabVIEW"* :

3.2.1 Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras dilakukan untuk mengetahui perangkat keras yang dibutuhkan selama penelitian ini. Perangkat keras yang dibutuhkan adalah :

1. PC (*Personal Computer*).
2. NI myRIO-1900.

3. *Access Point Wi-Fi (Wifi Tethering dari ponsel Android).*

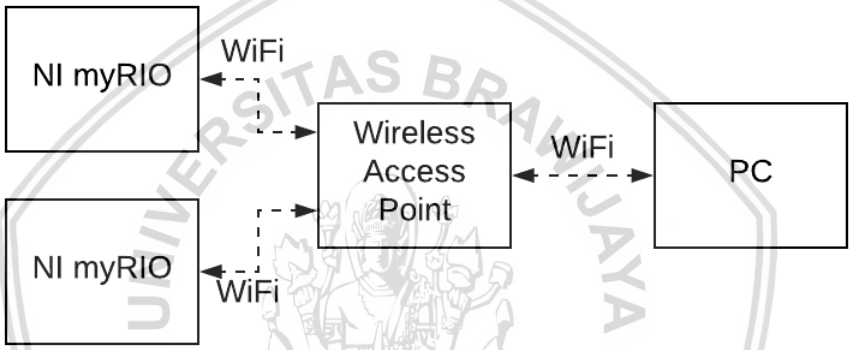
3.2.2 Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak dilakukan untuk mengetahui perangkat lunak yang dibutuhkan selama penelitian ini. Perangkat lunak yang dibutuhkan adalah :

- NI LabVIEW.

3.3 Perancangan Sistem

Tahap perancangan sistem merupakan perancangan dari arsitektur dan langkah kerja dari sistem, peneliti membuat topologi perancangan sistem seperti pada Gambar 3.2.



Gambar 3.2 Topologi Perancangan Sistem

Perangkat *embedded* NI myRIO akan terhubung dengan jaringan *Local Area Network* atau *WiFi* yang sama dengan PC. Pada NI myRIO akan melakukan pendeteksi *device* lain secara otomatis dan data *device status* yang terdapat pada NI myRIO dapat diambil oleh PC yang dikenali. Pada PC juga dapat melakukan pendeteksian perangkat lain secara otomatis dan dapat menerima data dari semua myRIO yang dikenali.

3.4 Implementasi

Tahap implementasi mengacu pada hasil perancangan sistem yang telah dilakukan. Implementasi dimulai dari spesifikasi sistem di mana dalam tahap ini memuat penjabaran dari spesifikasi perangkat keras, spesifikasi perangkat lunak yang digunakan dalam sistem ini.

Implementasi perangkat keras untuk penelitian ini menggunakan PC maupun laptop untuk menjadi *host,* lebih dari satu perangkat myRIO untuk meneliti fungsi *multi-devices,* dan *wireless access point* atau gawai *Android* yang memiliki fitur *WiFi tethering* sebagai media komunikasinya. Alamat IP yang didapatkan PC maupun myRIO tetap didapatkan secara otomatis dari DHCP.

Implementasi perangkat lunak untuk penelitian ini akan menerapkan mekanisme sistem *pervasive* pada PC dan myRIO agar dapat mengenali satu sama lain. Kemudian pada PC akan diprogram menjadi *host* untuk menerima data yang dikirim oleh semua myRIO. Sedangkan pada myRIO akan diprogram untuk menjadi *security box* yang akan mengirimkan data *device status*-nya ke PC yang dikenali.

3.5 Pengujian dan Analisis

Pengujian dan analisis pada penelitian ini dilakukan agar sistem dapat menunjukkan bahwa mampu untuk bekerja sesuai dengan tujuan beserta spesifikasi dan kebutuhannya. Pengujian fungsionalitas dilakukan dengan metode *State-based testing* untuk menguji sistem apakah transisi tiap *state* yang ada pada program telah berjalan sesuai dengan rancangan serta pengujian *delay* komunikasi antara PC dengan myRIO. Pengujian dilakukan dengan menerapkan langsung pada PC dan perangkat myRIO yang telah terhubung pada jaringan yang sama. Pengujian non-fungsional dilakukan dengan menguji PC dan myRIO dapat mengeksekusi program LabVIEW serta menguji PC dengan myRIO telah terhubung pada jaringan yang sama.

Apabila hasil dari pengujian telah memenuhi tujuan dari penelitian, maka berlanjut ke proses pengambilan kesimpulan dan saran. Sedangkan apabila hasil dari pengujian tidak memenuhi tujuan dari penelitian, maka kembali ke proses Perancangan sistem.

3.6 Kesimpulan dan saran

Kesimpulan didapatkan setelah melakukan desain dan perancangan, implementasi, pengujian sistem dan analisis sistem. Kesimpulan diambil berdasarkan hasil pengujian sistem dan analisis sistem yang dibuat. Isi dari kesimpulan diharapkan menjadi acuan pada penelitian lain untuk mengembangkan penelitian ini. Selain itu pada akhir penulisan terdapat saran yang bertujuan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk pengembangan sistem selanjutnya.

BAB 4 REKAYASA KEBUTUHAN SISTEM

4.1 Gambaran Umum Sistem

Sistem *"Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW*" menggunakan perangkat NI myRIO dan *Personal Computer. Personal Computer* digunakan sebagai pengontrol dan penerima data yang dikirimkan oleh myRIO. myRIO bertindak sebagai simulasi dari *Security Box* dan datanya dikirimkan ke PC yang telah terdaftar di myRIO. Penerapan UDP *Pervasive* pada sistem ini adalah perangkat PC dan myRIO dapat mengenali semua perangkat yang berada dalam jaringan yang sama dengan otomatis.

Pada penelitian ini menggunakan 1 PC dan 2 myRIO. PC dapat mengenali kedua myRIO yang berada pada jaringan serta menampilkan status perangkat myRIO yang diterima dengan rutin. PC juga dapat mengontrol status *locked* maupun *unlocked* dari myRIO. Untuk perangkat myRIO juga dapat mengenali PC serta myRIO yang lain. Setiap 5 detik sekali myRIO mengirimkan status perangkatnya ke PC.

4.2 Analisis Kebutuhan

Analisis kebutuhan sistem dilakukan untuk mengidentifikasi kebutuhan perangkat keras, perangkat lunak, dan fungsional dari penelitian "Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW". Berikut adalah rincian dari kebutuhan dari penelitian ini :

4.2.1 Kebutuhan Fungsional

Dengan adanya kebutuhan perangkat keras dan perangkat lunak, diharapkan penelitian "Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW" dapat mejalankan kebutuhan fungsional yang dirancang. Berikut adalah kebutuhan fungsional dari penelitian ini :

1. ***Personal Computer dapat mendeteksi otomatis perangkat lain.***

   Personal Computer dapat mengenali alamat IP dari perangkat lain yang berada satu jaringan *Wireless Local Area Network* yang sama secara otomatis.

2. ***Perangkat myRIO dapat mendeteksi otomatis perangkat lain.***

   Perangkat myRIO dapat mengenali alamat IP dari perangkat lain yang berada satu jaringan *Wireless Local Area Network* yang sama secara otomatis.

3. ***Perangkat myRIO dapat bertindak sebagai client dari Security Box.***

   Perangkat myRIO diprogram seolah-olah perangkat tersebut merupakan salah satu Security Box dan mengirimkan statusnya kepada PC yang tersedia di jaringa. Apabila myRIO tidak mendeteksi perlakuan yang

tidak diharapkan, maka status dari myRIO tetap *Safe.* Apabila mendeteksi gerakan atau tombol dari myRIO ditekan, maka status dari myRIO menjadi *Unsafe (Motion Detected* atau *Button Detected).* Untuk mengubah status *Unsafe* menjadi *Safe* lagi membutuhkan pengguna PC untuk meng-*unlock* perangkat myRIO dengan memasukkan *username* dan *password* dari perangkat myRIO yang telah didefinisikan sebelumnya.

4. ***Personal Computer dapat bertindak sebagai host dari Security Box***

   Personal Computer diprogram sebagai *host* dari fungsi *Security Box* perangkat myRIO. PC dapat menerima semua status dari tiap perangkat myRIO serta dapat mengatur status dari perangkat myRIO apabila dibutuhkan.

4.2.2 Kebutuhan Non-Fungsional

Analisis Kebutuhan Non-Fungsional meliputi penjabaran dari analisis kebutuhan perangkat keras dan perangkat lunak yang akan digunakan dalam penelitian ini.

4.2.2.1 Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras dilakukan untuk mengetahui perangkat keras yang dibutuhkan selama penelitian "Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW". Perangkat keras yang dibutuhkan adalah:

1. ***Personal Computer*.**

   Personal Computer digunakan untuk memprogram aplikasi berbasis LabVIEW untuk PC itu sendiri dan untuk perangkat myRIO yang digunakan untuk penelitian ini. Personal Computer juga digunakan sebagai pengontrol myRIO karena memiliki Front Panel dari program LabVIEW dan juga salah satu perangkat yang diteliti sebagai *node* dalam satu jaringan yang sama dengan myRIO.

2. **National Instruments myRIO.**

   NI myRIO merupakan perangkat embedded dan Real-Time System yang dikembangkan oleh National Instruments. Perangkat ini digunakan pada penelitian ini karena memiliki fitur sensor akselerometer untuk mendeteksi gerakan, push button untuk mendeteksi pengoperasian, dan memiliki Wireless Radio agar dapat terhubung dengan Wireless Local Area Network. Perangkat ini mendukung untuk pemrograman paralel sehingga dapat diterapkan pada penelitian ini. Perangkat myRIO juga dianggap sebagai node dalam satu jaringan yang sama dengan PC.

3. **Wireless Access Point.**

Wireless Access Point digunakan sebagai media untuk menghubungkan sebuah PC dengan kedua perangkat myRIO. Semua perangkat tetap mendapatkan alamat IP dengan otomatis dari DHCP.

4.2.2.2 Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak dilakukan untuk mengetahui perangkat lunak yang dibutuhkan selama penelitian "Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW". Perangkat lunak yang dibutuhkan adalah :

(1) National Instruments LabVIEW

LabVIEW adalah perangkat lunak yang digunakan untuk platform desain sistem dan menggunakan *dataflow programming.* LabVIEW digunakan pada penelitian ini karena mendukung untuk pemrograman dengan desain sistem berbasiskan *state-machine* dan pemrograman paralel. Serta LabVIEW memiliki kompatibilitas penuh untuk memprogram perangkat NI myRIO. LabVIEW juga dapat untuk membuat program yang dapat dijalankan pada PC.

4.3 Batasan Sistem

Pada penelitian *"Implementasi Rancangan Protokol UDP Pervasive Multi Device untuk Perangkat myRIO Berbasis LabVIEW*" memiliki beberapa batasan yang diterapkan. Berikut adalah batasan sistem yang diterapkan :

1. Sistem diimplementasikan pada perangkat embedded myRIO dengan bahasa pemrograman LabVIEW dan PC (*Personal Computer*) yang dapat menjalankan program berbasis bahasa pemrograman LabVIEW.
2. Pengujian menggunakan tipe data *string* yang direpresentasikan sebagai perintah atau data status.
3. Perangkat yang diuji hanya berada dalam satu jaringan *Wireless Local Area Network* yang sama.
4. Pertukaran data antar perangkat menggunakan protokol UDP.
5. Untuk simulasi pengujian, perangkat myRIO bertindak seperti *Security Box.*
6. Hanya terdapat 1 PC yang dapat menjalankan program dalam satu waktu.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada BAB ini akan menjelaskan perancangan dan implementasi dari sistem yang meliputi perancangan perangkat keras, perancangan perangkat lunak, implementasi perangkat keras, dan implementasi perangkat lunak.

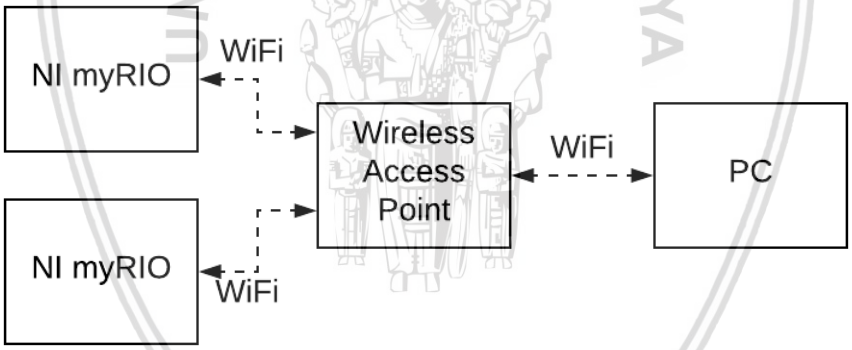
5.1 Perancangan Sistem

Pada subbab ini akan menjelaskan proses perancangan perangkat keras yang meliputi diagram blok dan perancangan perangkat lunak yang meliputi *state machine* dari *personal computer* dan NI myRIO.

5.1.1 Perancangan Perangkat Keras

5.1.1.1 Topologi Perangkat

Perangkat keras pada sistem ini meliputi *personal computer* yang bertindak sebagai host dan juga dua perangkat NI myRIO yang bertindak sebagai client. *Personal computer* bertugas untuk mengontrol dan menerima data dari perangkat NI myRIO. Sedangkan NI myRIO bertugas untuk mengirimkan data ke PC dan diprogram untuk menyimulasikan perangkat *Security Box*.



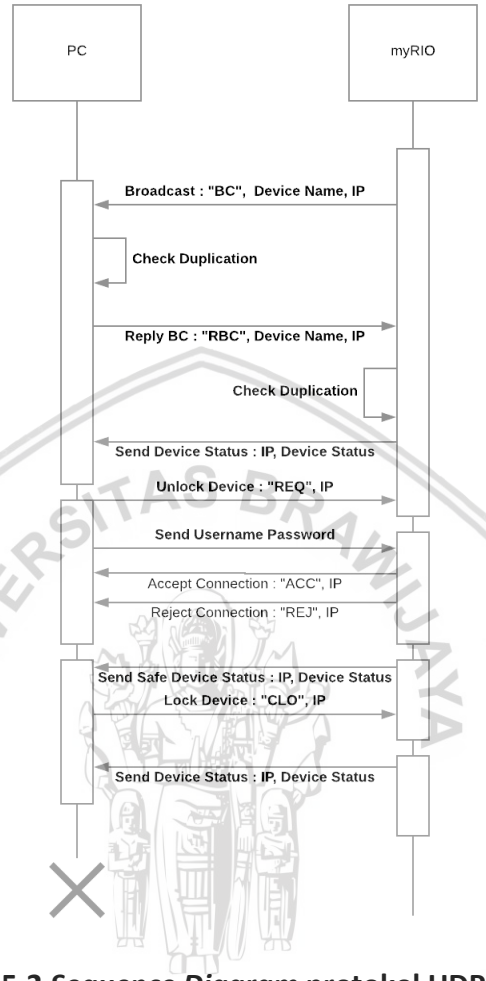
Gambar 5.1 Topologi Sistem

Ketiga perangkat tersebut dihubungkan dengan media komunikasi WiFi yang disediakan oleh *Wireless Access Point* seperti yang diilustrasikan oleh Gambar 5.1. Alamat IP yang digunakan oleh ketiga perangkat tetap didapatkan otomatis dari DHCP. Perangkat dari *Wireless Access Point* dapat disubstitusikan dengan ponsel yang memiliki fitur *WiFi Tethering* untuk memudahkan pengujian di mana saja.

Perangkat NI myRIO membutuhkan daya yang hanya bisa didapatkan dari menghubungkan adaptornya ke stop kontak. Sedangkan untuk Wireless Access Point dapat disubstitusikan dengan gawai *Android* sehingga dapat aktif menggunakan baterai. Untuk PC juga dapat menggunakan laptop yang dapat aktif menggunakan baterai.

5.1.2 Perancangan Perangkat Lunak

5.1.2.1 Sequence Diagram



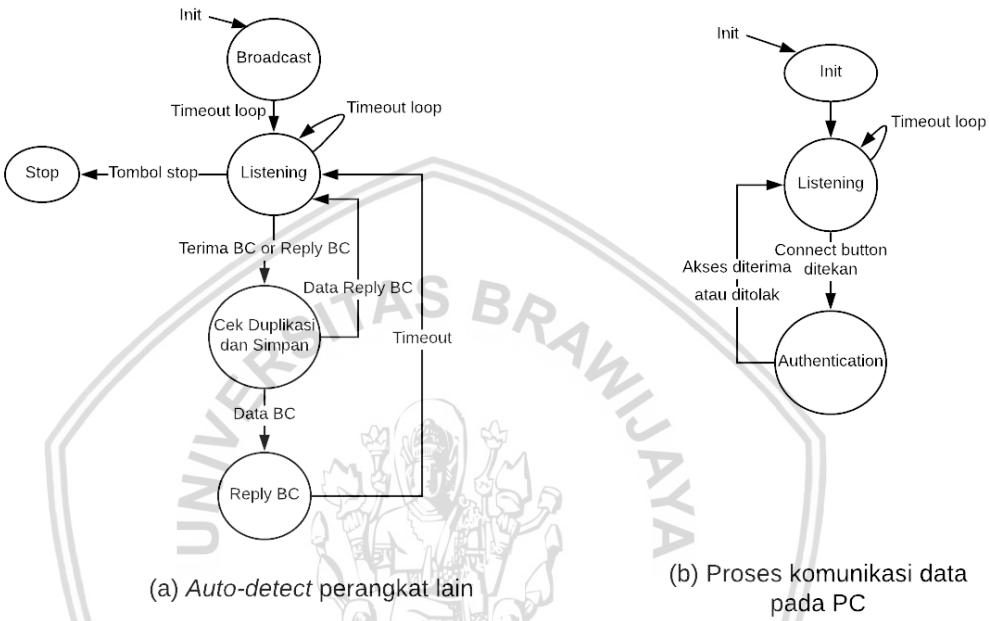
Gambar 5.2 *Sequence Diagram* protokol UDP pervasif

Gambar 5.2 merupakan *sequence diagram* komunikasi data dari myRIO kepada PC. Berdasarkan diagram tersebut diasumsikan program PC telah dieksekusi dan telah memasuki *state "Listening".* Ketika myRIO pertama kali diaktifkan akan mengirimkan *broadcast* yang membawa nama perangkat dan alamat IP agar perangkat lain yang menerima *broadcast* tersebut dapat mengenali myRIO tersebut. Perangkat lain atau PC yang menerima *broadcast* tersebut akan melakukan cek duplikasi dan menyimpan nama dan alamat dari myRIO pengirim *broadcast.* Selanjutnya PC atau perangkat lain penerima *broadcast* mengirimkan pesan balasan yang membawa nama perangkat dan alamat IP agar myRIO pengirim *broadcast* dapat mengenali perangkat lain serta melakukan cek duplikasi. Setelah itu semua perangkat yang aktif pada jaringan dapat saling mengenali nama perangkat serta alamat IP-nya.

Perangkat myRIO mengirimkan *device status* kepada PC yang dikenalinya. PC dapat melakukan *unlock* myRIO dengan mengirimkan *request* dan mengirimkan *username* dan *password* dari myRIO. Perangkat myRIO akan melakukan

autentikasi dan mengirimkan hasil autentikasi kepada PC. Ketika data autentikasi benar maka myRIO ter-*unlock* dan mengirimkan *status* aman apapun yang terjadi. PC dapat me-*lock* myRIO kembali dengan mengirimkan pesan "CLO" dan myRIO kembali menjadi mode *locked*.

5.1.2.2 State Machine Personal Computer



Gambar 5.3 State diagram pada PC

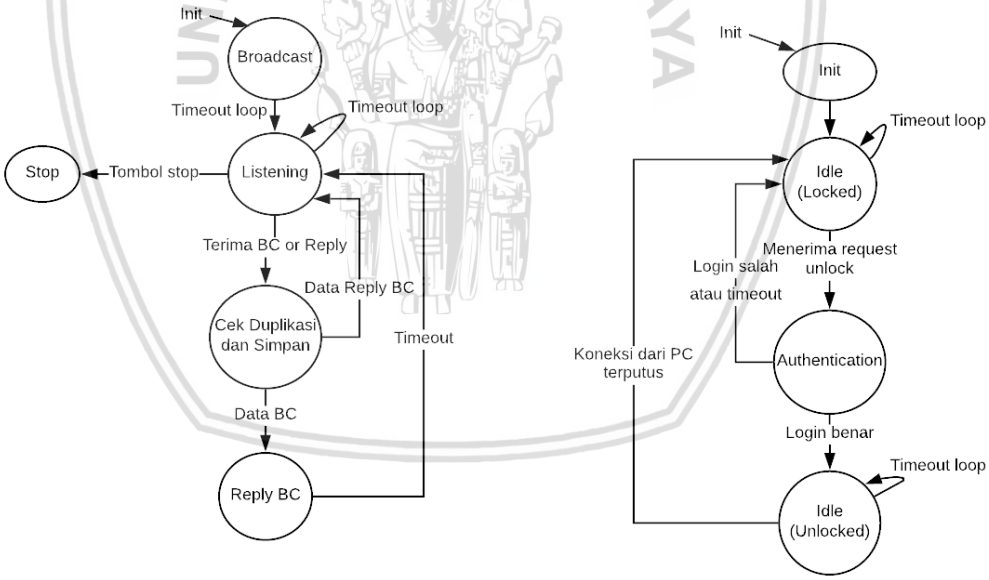
Program untuk *Personal Computer* dirancang menggunakan metode *Finite State Machine* seperti yang ditunjukkan pada Gambar 5.3. Program yang dirancang memiliki 2 *state* diagram yang dapat berjalan secara bersamaan. Untuk *state* diagram bagian (a) berfungsi untuk mendeteksi perangkat lain yang ada pada jaringan yang sama. Ketika program pertama dijalankan, perangkat akan melakukan *broadcast*. Kemudian program berubah ke *state listening*. Apabila selama *listening* menerima *broadcast* atau ACK dari perangkat lain, maka berpindah ke *state* cek duplikasi. Setelah itu apabila data yang diterima adalah *broadcast*, maka program berpindah ke *state* kirim ACK ke pengirim dari *broadcast* lalu kembali ke *state listening*. Apabila data yang diterima adalah ACK maka program kembali ke *state listening*.

Untuk *state* diagram bagian (b) digunakan untuk pertukaran data antara kedua perangkat apabila telah saling mengenal. Pertama program melakukan *inisialisasi open port* dan langsung ke kondisi *listening*. Selama kondisi *listening,* PC dapat menerima data berupa *device status* yang didapatkan dari perangkat myRIO yang telah mengenali PC. Pengguna dapat menekan tombol *Connect* yang digunakan untuk mematikan maupun menghidupkan fungsi *security box* yang digunakan myRIO. Ketika PC terhubung dengan myRIO, maka myRIO akan

mematikan fungsi *security box*-nya. Sebaliknya ketika myRIO tidak terhubung dengan PC, maka myRIO mengaktifkan fungsi *security box*-nya. Apabila pengguna menekan *Connect* ketika tidak terhubung dengan myRIO, program pindah ke kondisi *Authentication.* Pada kondisi ini program membuka port khusus untuk *authentication* dan akan meminta pengguna untuk memasukkan *username* dan *password* dari perangkat myRIO yang telah dipilih pada kondisi sebelumnya. Apabila *username* dan *password* tersebut diterima, maka perangkat myRIO akan mematikan fungsi *security box* dan PC akan dianggap terhubung dengan myRIO pada kondisi *listening.* Apabila *username* dan *password* ditolak atau mengalami *timeout*, PC tetap kembali ke kondisi *listening* dengan menampilkan peringatan bahwa komunikasi ditolak.

Ketika PC sedang terhubung dengan myRIO pada kondisi *listening*, PC tidak dapat memilih myRIO yang lain untuk dihubungkan lagi dan hanya dapat memutuskan hubungan myRIO yang sedang terhubung. Apabila pengguna menekan tombol *Connect* ketika terhubung dengan myRIO, maka PC memutuskan hubungan dengan myRIO dan myRIO kembali mengaktifkan fungsi *security box.* PC tetap berada pada kondisi *listening.*

5.1.2.3 State Machine NI myRIO



(a) *Auto-detect* perangkat lain

(b) Proses komunikasi data pada myRIO

Gambar 5.4 State diagram myRIO

Program untuk perangkat myRIO juga dirancang menggunakan metode *Finite State Machine.* Program yang dirancang juga memiliki 2 *state diagram* yang dapat berjalan bersamaan. Untuk *state* diagram bagian (a) berfungsi untuk

mendeteksi perangkat lain yang ada pada jaringan yang sama. *State diagram* (a) identik dengan *state diagram* (a) yang dirancang untuk PC karena menggunakan metode yang sama dalam mengenali perangkat lain yang berada pada jaringan yang sama.

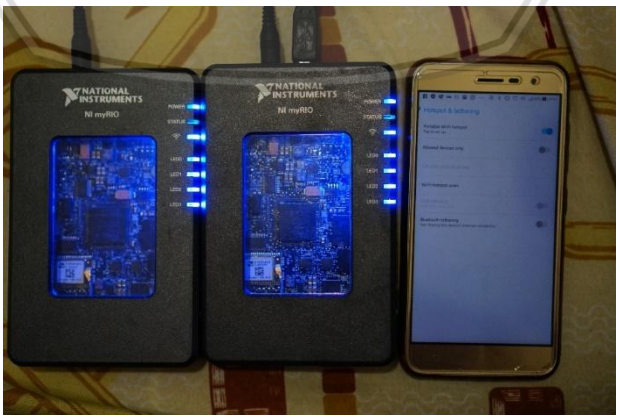
Untuk *state diagram* (b) digunakan untuk pertukaran data antara kedua perangkat apabila telah saling mengenal. Pertama program melakukan *inisialisasi open port* dan langsung ke kondisi *idle (locked).* Pada kondisi *idle (locked),* perangkat myRIO mengirimkan data yang berupa *device status* dari perangkat ini ke PC dalam jeda waktu yang telah ditentukan. *Device status* dapat berubah-ubah apabila perangkat myRIO mendeteksi adanya interaksi seperti tombol fisik ditekan maupun perangkat digoyangkan. Untuk me-*reset device status,* myRIO perlu di-*unlock* terlebih dahulu. Apabila PC meminta untuk terhubung maka program menjadi kondisi *Authentication.* Pada kondisi *Authentication,* program menunggu data *username* dan *password* yang diberikan oleh PC. Apabila data tersebut benar maka program menjadi kondisi *Idle (unlocked).* Pada kondisi *Idle (unlocked), device status* dari myRIO akan tetap *safe* meskipun mendapat interaksi dari pengguna dan mengirimkan *device* statusnya ke PC dalam jeda waktu yang ditentukan. Ketika PC memutuskan hubungannya dengan myRIO, maka kondisi dari myRIO kembali menjadi *Idle (locked).*

5.2 Implementasi Sistem

Pada subbab ini akan menjelaskan mengenai implementasi perangkat keras dan implementasi perangkat lunak dari sistem yang telah dirancang pada subbab sebelumnya.

5.2.1 Implementasi Perangkat Keras

Menurut dengan perancangan perangkat keras, dua buah perangkat myRIO terhubung dengan PC secara nirkabel melalui media WiFi yang disediakan oleh *Wireless Access Point* seperti yang diilustrasikan pada Gambar 5.5.



Gambar 5.5 Implementasi perangkat keras

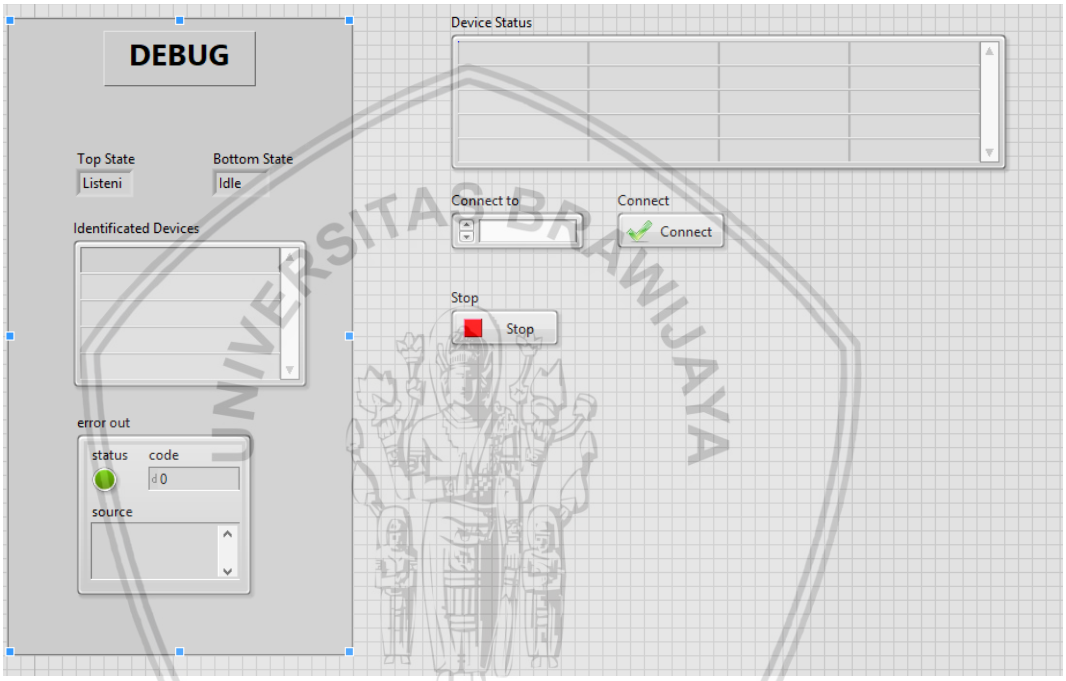
Kedua perangkat myRIO hanya membutuhkan daya yang didapatkan dari adaptor serta menghubungkannya pada *access point* yang juga terhubung dengan PC. PC yang digunakan dapat disubstitusikan dengan laptop sehingga

dapat menggunakan daya dari baterai. Sedangkan untuk perangkat *wireless access point* dapat menggunakan gawai yang telah mendukung fitur *WiFi Tethering*. Alamat IP yang didapatkan oleh PC dan myRIO diperoleh secara otomatis dari DHCP.

5.2.2 Implementasi Perangkat Lunak

5.2.2.1 Implementasi Program pada PC

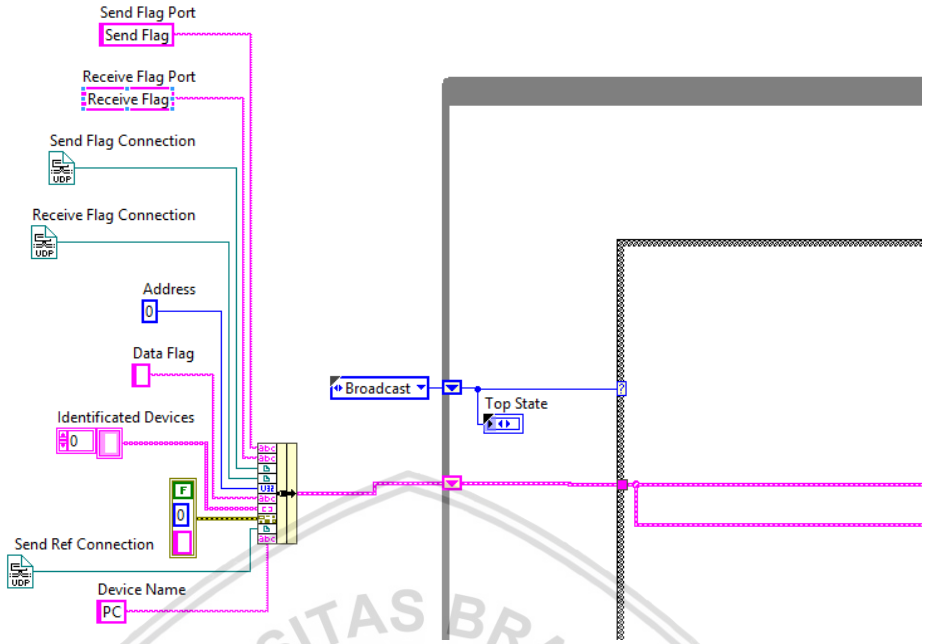
Pada pemrograman LabVIEW, pengembang dapat memprogram *Front Panel* untuk antarmuka dan *Block Diagram* sebagai algoritma dari program. *Front Panel* yang diimplementasikan diilustrasikan pada Gambar 5.6.



Gambar 5.6 Tampilan *Front Panel* pada PC

Berdasarkan Gambar 5.6 terdapat bagian *Debug* yang digunakan untuk proses *debugging* selama pengembangan seperti mengecek *state* pada suatu waktu, perangkat yang terdeteksi, dan deskripsi *error* jika muncul. Lalu terdapat *device status* yang berisi nama perangkat yang terdeteksi, *IP address* dari perangkat, status perangkat tersebut, serta waktu dari perangkat terakhir di-*update.* Selain itu terdapat *Text Ring* yang berfungsi untuk memilih perangkat yang akan dihubungkan dan *Connect button* untuk membuat perangkat yang dipilih *Text Ring* menjadi ke kondisi *unlocked* dan *locked.* Terakhir terdapat tombol Stop untuk menghentikan program.

Terdapat 2 *state machine* yang diimplementasikan pada satu *while loop*, yaitu *State machine* untuk mendeteksi perangkat lain secara otomatis dan *state machine* untuk komunikasi data pada PC. Setiap *state machine* memiliki inisialisasi *constant* yang akan di-*bundle* menjadi cluster.

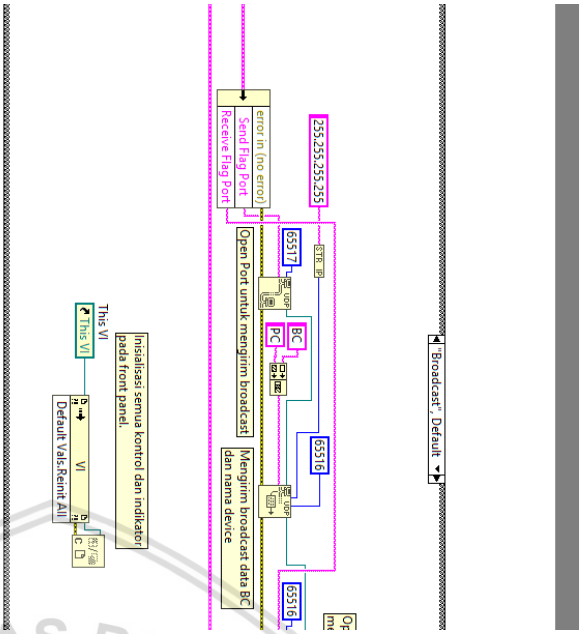


Gambar 5.7 Inisialisasi *constant* yang akan digunakan pada *auto-detect state machine*

Untuk detail mengenai *constant* yang digunakan pada Gambar 5.7 akan dijelaskan pada Tabel 5.1.

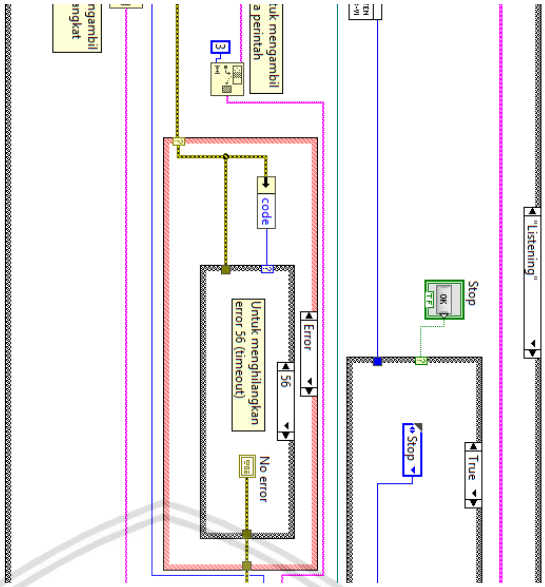
Tabel 5.1 Penjelasan mengenai Inisialisasi *auto-detect state machine*

Nama Blok	Tipe Data	Fungsi
Send Flag Port	*String*	*Service name* pada blok UDP Open
Receive Flag Port	*String*	*Service name* pada blok UDP Open
Rend Flag Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP
Receive Flag Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP
Address	*Integer*	Menyimpan alamat IP dari blok UDP
Data Flag	*String*	Menyimpan perintah yang dipakai pada proses *auto detect* perangkat
Identificated Devices	*Array of String*	Menyimpan nama perangkat yang terdeteksi pada jaringan yang sama
Error in	*Error Constant*	Inisialisasi keadaan *error*
Send Ref Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP
Device Name	*String*	Menyimpan nama dari perangkat



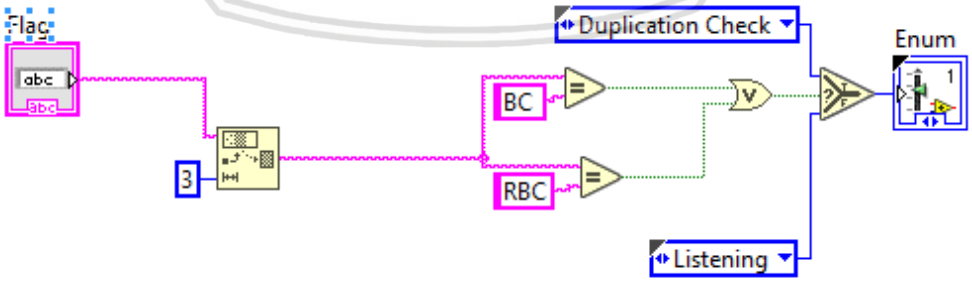
Gambar 5.8 *State Broadcast* pada PC

Pada *state machine* yang pertama dimulai dengan *state Broadcast*. Berdasarkan Gambar 5.8 pada *state* ini dimulai dengan *open port* 65517 dengan *service name* Send Flag. Port ini digunakan untuk mengirim data yang terkait dengan pendeteksian perangkat otomatis seperti BC atau ACK. Kemudian melalui *port* tersebut dilakukan pengiriman data "BC PC" ke 255.255.255.255 port 65516 yang berarti *broadcast* ke semua perangkat pada jaringan yang sama. *String* "BC PC" berarti broadcast oleh PC. Lalu dilakukan *open port* 65516 yang digunakan untuk menerima data *broadcast* maupun ACK. Pembukaan *port* dilakukan setelah pengiriman *broadcast* bertujuan agar data dari *broadcast* tersebut tidak terkirim ke perangkat itu sendiri dan mencegah duplikasi data. Selain itu dilakukan *open port* 65512 yang digunakan untuk pengenalan PC oleh perangkat myRIO dan blok *Invoke Node* yang bertujuan untuk mengembalikan semua elemen pada *front panel* kembali menjadi *default*. *State* selanjutnya adalah *state Listening.*



Gambar 5.9 *State Listening* pada PC

State Listening bertindak sebagai penerima data dari broadcast maupun ACK. Ketika PC menerima broadcast maka data dan pengirim data tersebut diolah pada *state ini.* Berdasarkan Gambar 5.9 program dimulai dari blok *UDP Read* yang berfungsi untuk menerima data dari protokol UDP pada *port* yang terhubung dengan *Receive Flag Connection* (port 65516) dengan timeout 100 (berarti 100 milidetik). Data yang diterima akan diolah dengan memisahkan *string* menjadi dua bagian, yaitu bagian *Data Flag* yang didapatkan dari 3 karakter pertama *string* dan bagian *Device Name* yang didapatkan dari karakter sisa yang dimulai dari karakter ke-4. Sebagai contoh terdapat data *string* yang berisi *"*BC PC". String tersebut dipisah menjadi "BC " sebagai *Data Flag* dan "PC" sebagai *Device Name.* Data Flag digunakan untuk menentukan *next state* dari *state Listening* dan *state Duplication Check.* Penentuan *next state* dari *state listening* diprogram pada sub-VI. Blok diagram dari sub-VI ini diilustrasikan pada Gambar 5.8.

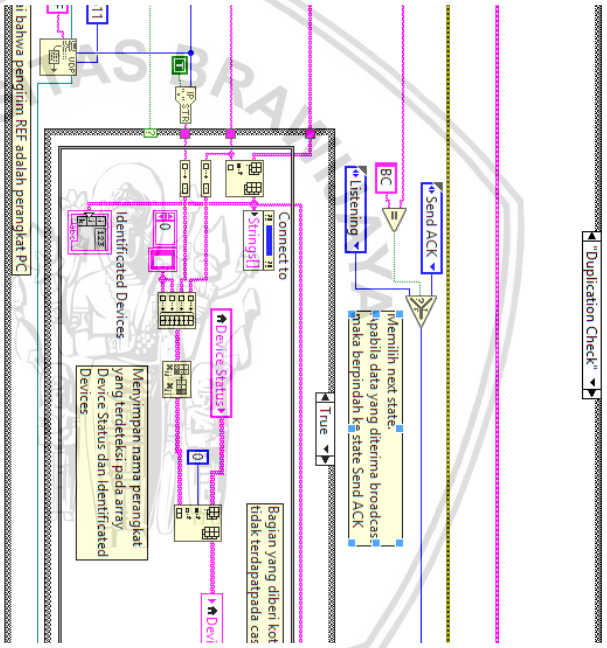


Gambar 5.10 Sub-VI untuk memilih *next state*

Berdasarkan Gambar 5.10, apabila *Data Flag* berisi "BC" atau "RBC" maka *next state* dari *state Listening* adalah *Duplication Check.* Selain itu *next state*nya tetap *Listening.*

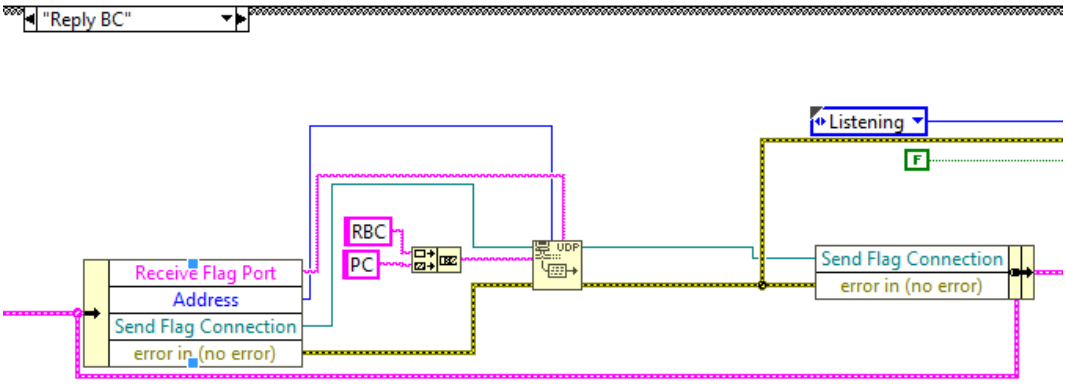
Pada *state Listening* terdapat *error handling* dengan meniadakan *error* 56 dengan tujuan selama *Listening* tidak mempermasalahkan adanya *timeout* apabila tidak ada data yang diterima dalam jangka waktu 100 milidetik. Pada *state Listening* juga pengguna dapat menghentikan program dengan menekan tombol Stop. Dengan mengaktifkan tombol Stop maka *next state*-nya menjadi ke *state Stop.*

Berdasarkan Gambar 5.11, *State* ini digunakan untuk pengecekan duplikasi untuk mencegah perangkat yang sama untuk disimpan lebih dari sekali. Nama perangkat yang didapatkan pada *state Listening* akan dicari pada *array Identificated Devices.* Apabila hasil pencarian bernilai kurang dari nol berarti nama perangkat yang dicari tidak ada dan alamat serta nama perangkat tersebut disimpan pada *array Device Status* yang terdapat pada *front panel.* Pada *state* ini juga akan mengirimkan data ”REF” kepada port 65511 yang bertujuan untuk memberi tahu bahwa pengirim “REF” adalah perangkat PC.



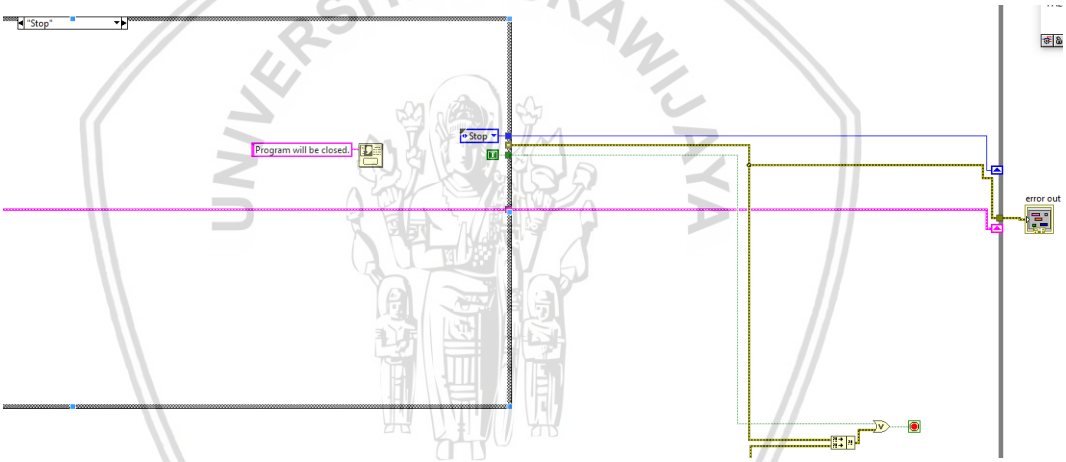
Gambar 5.11 *State Duplication Check* pada PC

Next state dari *state* ini tergantung pada *Data Flag* yang diterima pada *state Listening.* Apabila *Data Flag* berisi “BC ” maka *next state*-nya adalah *Reply BC.* Selain itu *next state*-nya kembali ke *state Listening*.



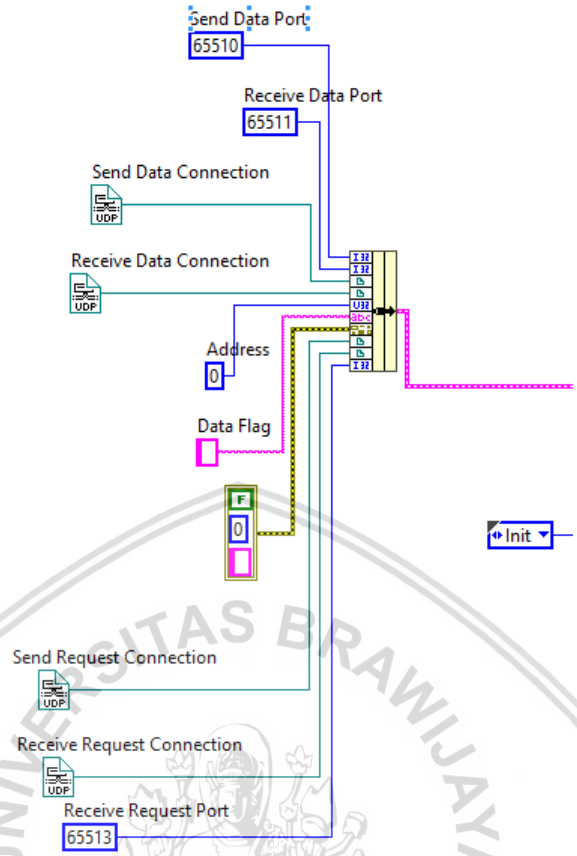
Gambar 5.12 *State Reply BC* pada PC

Selanjutnya terdapat *state Reply BC* seperti yang digambarkan pada Gambar 5.12. *State* ini akan mengirimkan *string* yang berisi gabungan dari "RBC" sebagai *Data Flag* dan "PC" sebagai *Device Name* kepada pengirim *broadcast. Next state* dari *state* ini adalah kembali ke *state* Listening.



Gambar 5.13 *State Stop* pada PC

Pada *state Stop* seperti yang digambarkan pada Gambar 5.13 akan memberikan *popup* yang menandakan program akan berhenti serta mengeluarkan *boolean* bernilai *true* yang terhubung dengan kondisi *"Stop if true"* dari *while loop.* Dengan menghentikan *while loop* maka program juga akan berhenti.



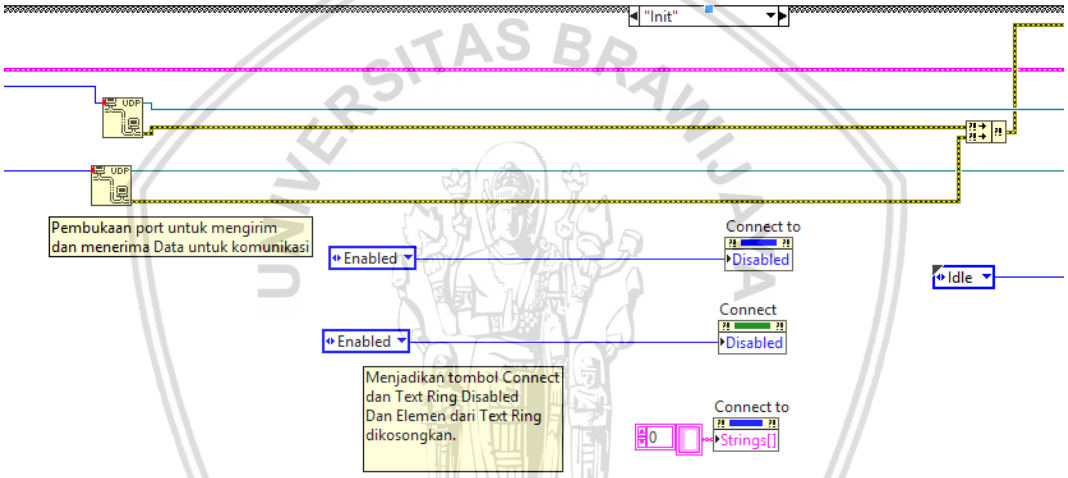
Gambar 5.14 Inisialisasi *constant* untuk *state machine* proses komunikasi data

Untuk *state machine* yang kedua pada PC berfungsi untuk proses komunikasi data ke perangkat myRIO. Blok yang digunakan untuk inisialisasi tersebut digambarkan pada Gambar 5.14 dan detail mengenai fungsinya dijelaskan pada Tabel 5.2.

Tabel 5.2 Penjelasan mengenai *inisialisasi state machine* pertukaran data.

Nama Blok	Tipe Data	Fungsi
Send Data Port	*Integer*	*Port* untuk mengirimkan data.
Receive Data Port	*String*	*Port* untuk menerima data.
Rend Data Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP menerima data.
Receive Data Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP mengirim data.
Address	*Integer*	Menyimpan alamat IP dari blok UDP.

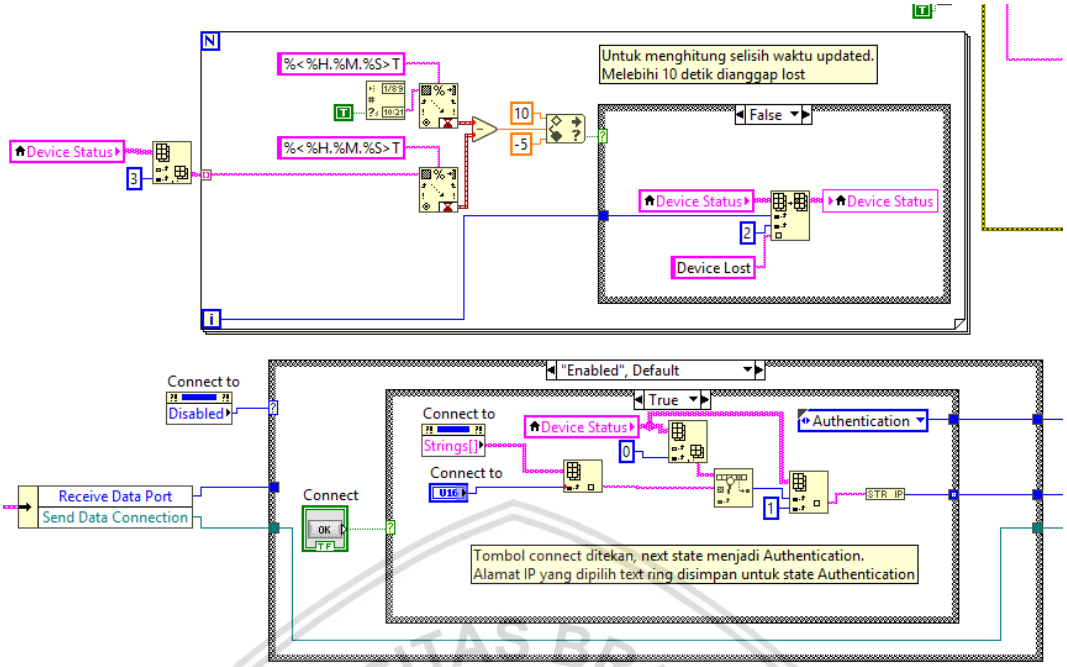
Data Flag	*String*	Menyimpan perintah yang dipakai pada proses pertukaran data
Send Request Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP mengirim *request.*
Error in	*Error Constant*	Inisialisasi keadaan *error*
Receive Request Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP menerima *request.*
Receive Request Port	*Integer*	*Port* untuk menerima *request.*



Gambar 5.15 *State Init* pada PC

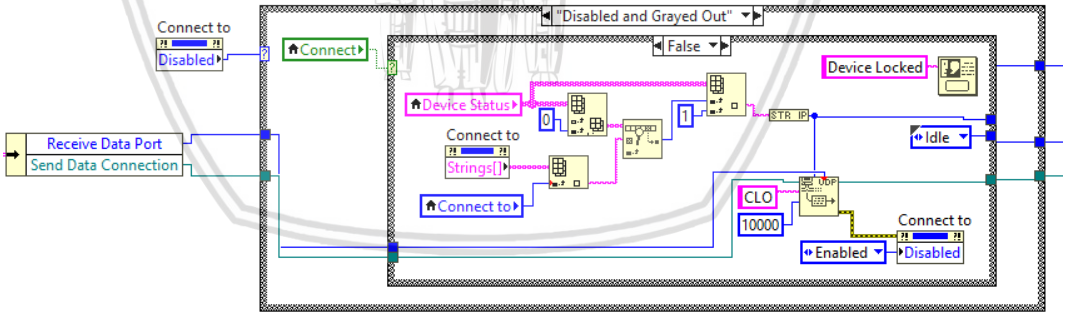
State Init digunakan untuk membuka *port* untuk mengirim dan menerima data untuk komunikasi, serta mengaktifkan fungsi dari tombol *connect* dan *text ring. Next state* nya adalah *state Idle.*

Pada *state Idle,* PC dapat melakukan perhitungan selisih waktu terkini dengan waktu data status terakhir diterima PC dan PC dapat memilih perangkat lain yang akan dihubungkan seperti yang digambarkan oleh Gambar 5.16. Untuk dapat menghitung selisih waktu seluruh perangkat yang telah terdeteksi diperlukan *for loop* dengan fitur indexing sehingga dilakukan perulangan sejumlah elemen dari *array Device Status.* Apabila hasil dari perhitungan selisih lebih dari 10 detik maupun minus 5 detik, maka perangkat dianggap *lost.* Ketika PC dalam keadaan tidak terhubung dengan perangkat lain, maka pengguna dapat memilih perangkat lain yang akan dihubungkan melalui *text ring* dan menekan tombol Connect. Ketika tombol ditekan, maka program berpindah ke *state Authentication* dengan menyimpan alamat dari perangkat yang akan dihubungkan ke *shift register*.

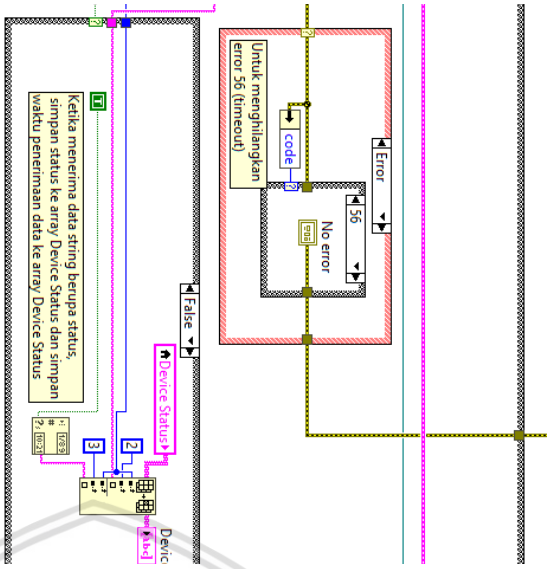


Gambar 5.16 *State Idle* bagian pertama pada PC

Gambar 5.17 merupakan nilai *case* lain dari *case* dengan kondisi *property node text ring.* Ketika PC dalam keadaan terhubung dengan myRIO, *Text Ring* dalam keadaan *locked* yang diilustrasikan menjadi *Grayed Out.* Pada keadaan ini apabila tombol Connect dinonaktifkan maka Text Ring kembali *diunlock* dan mengirimkan *string* "CLO" ke perangkat yang sedang dihubungkan untuk menutup komunikasi dengan perangkat.

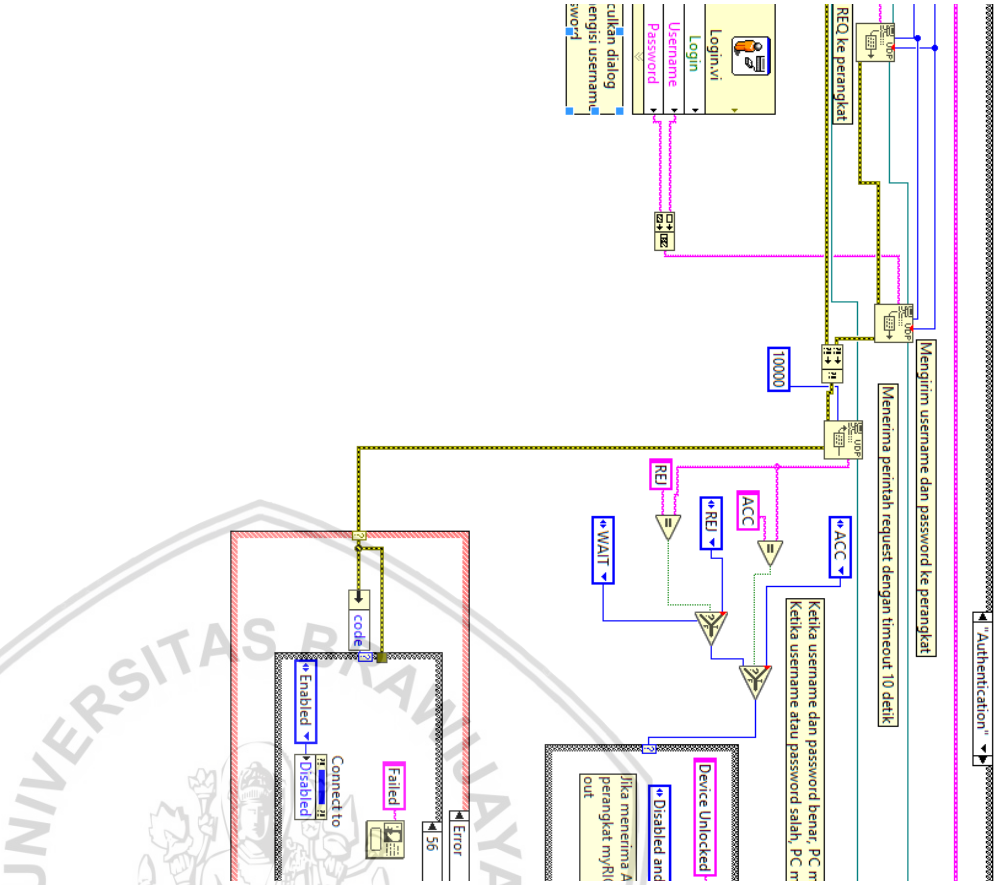


Gambar 5.17 *State Idle* bagian kedua pada PC



Gambar 5.18 State Idle bagian ketiga pada PC

Pada *state Idle* juga dapat menerima data *string* yang berupa *Status* dari perangkat myRIO seperti yang digambarkan pada Gambar 5.18. Alamat pengirim data status akan dicari pada *array Device Status* dan nilai indexnya digunakan untuk menyimpan *string* status dan waktu penerimaan data pada baris yang tepat pada *array Device Status.* Pada state ini terdapat *error handling* untuk mengabaikan *error* yang muncul akibat *timeout* ketika tidak ada data yang diterima.

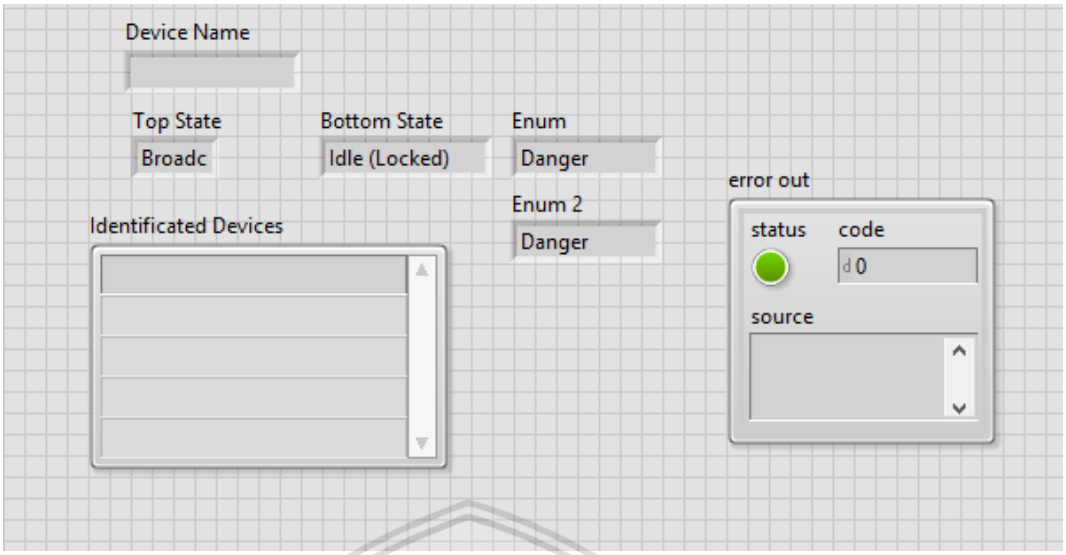


Gambar 5.19 *State Authentication* pada PC

State Authentication berfungsi untuk keperluan *autentikasi* perangkat myRIO dengan mengisikan *username* dan *password* yang tepat seperti yang diilustrasikan pada Gambar 5.19. Pertama program membuka *port* yuang digunakan untuk keperluan autentikasi. Kemudian PC mengirimkan *string* "REQ" ke perangkat yang telah dipilih pada *Text Ring* di *state* sebelumnya. Kemudian pengguna PC memasukkan *username* dan *password* dari perangkat myRIO. *Username* dan *password* tersebut disatukan menjadi satu *string* dan dikirimkan ke perangkat myRIO. Kemudian PC menunggu balasan dari perangkat myRIO. Ketika PC menerima *string* "ACC" menandakan data yang *diinputkan* benar dan myRIO dalam keadaan *unlocked* dan terhubung dengan PC serta *text ring* menjadi *grayed out*. Ketika menerima *string* "REJ" menandakan data yang *diinputkan* salah dan tombol *Connect* serta *text ring* dikembalikan ke keadaan semula. *Next state* dari *state* ini adalah kembali ke *state Idle.*

5.2.2.2 Implementasi Program pada myRIO

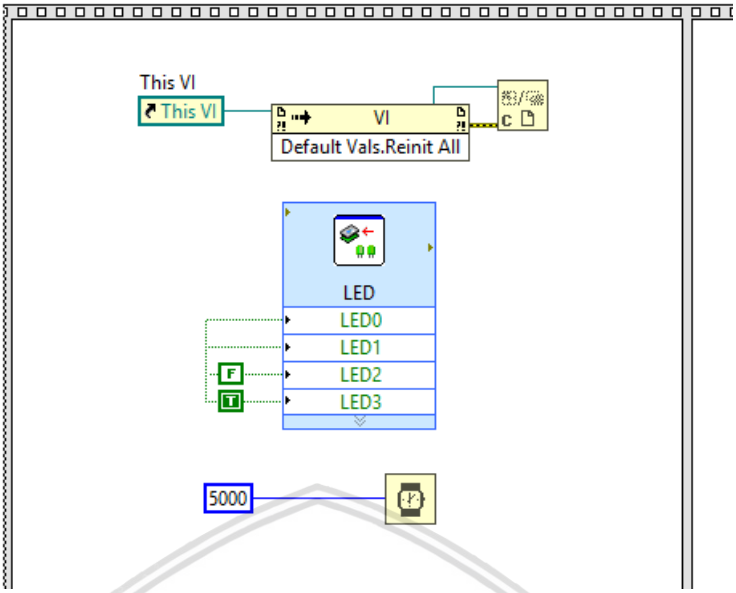
Implementasi program untuk perangkat myRIO juga menggunakan dua *state machine* seperti yang diimplementasikan pada PC, yaitu *state machine* untuk mendeteksi perangkat otomatis dan *state machine* untuk komunikasi data. Sedangkan untuk *Front Panel* pada myRIO hanya ditujukan untuk keperluan *debugging* dan tidak berinteraksi langsung dengan pengguna.



Gambar 5.20 Tampilan *Front Panel* pada myRIO

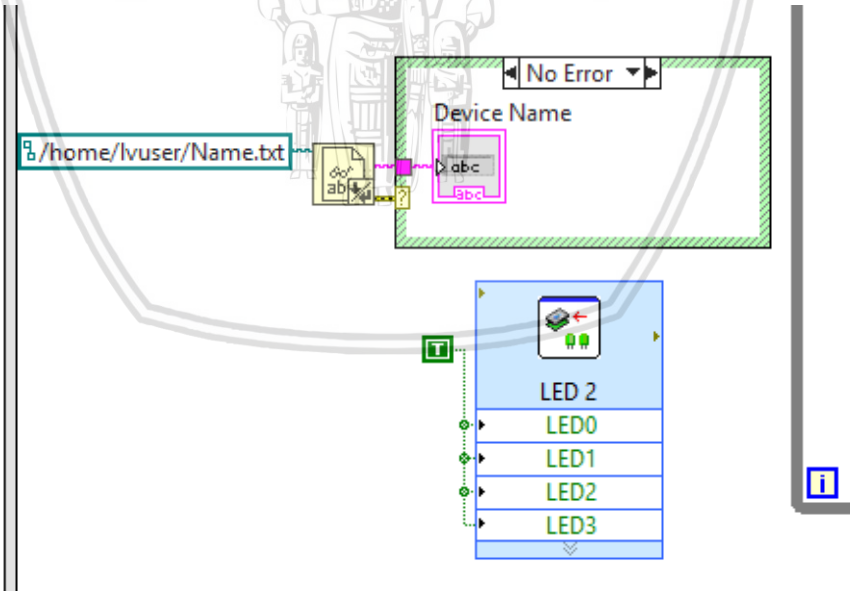
Berdasarkan Gambar 5.20, tampilan *Front Panel* memiliki indikator *Device Name* untuk menampilkan nama perangkat, *Top State* dan *Bottom State* untuk menampilkan *state* yang sedang aktif pada suatu waktu, *Enum* dan *Enum 2* untuk menampilkan keadaan status perangkat yang sedang aktif, *Identificated Devices* untuk menyimpan perangkat yang terlah terdeteksi, dan *error out* untuk menampilkan *error.*

Program untuk myRIO menggunakan struktur *state machine* yang berada di dalam *flat sequence frame* kedua agar perangkat myRIO memiliki waktu yang cukup untuk terhubung dengan *WiFi* sebelum program utama dieksekusi dengan memberikan *delay* pada *frame* pertama seperti yang digambarkan pada Gambar 5.21. Pada *frame* pertama juga dilakukan *inisialisasi front panel* kembali ke nilai *default* dan penggunaan LED untuk keperluan debugging.



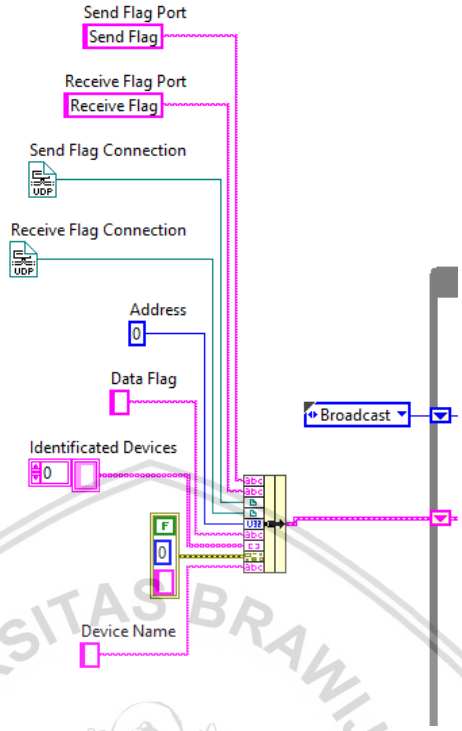
Gambar 5.21 *Frame* pertama pada *Flat Sequence*

Frame kedua pada *Flat Sequence* berisi program utama *while loop state machine.* Pada *frame* ini terdapat blok untuk membaca *text file* yang telah dibuat pengguna dan tersimpan pada penyimpanan myRIO. *Text file* ini digunakan sebagai *Device Name* yang menjadi identitas setiap perangkat myRIO. Selain itu juga terdapat blok LED untuk keperluan *debugging.* Blok membaca *text file* dan blok LED digambarkan pada Gambar 5.22.



Gambar 5.22 Blok untuk mendapatkan *Device Name* dan fungsi LED

Sama seperti yang diimplementasikan pada PC, pada myRIO juga membutuhkan *inisialisasi constant* yang akan digunakan untuk menyimpan data pada *shift register.* Blok yang digunakan *inisialisasi state machine* pendeteksi perangkat otomatis digambarkan pada Gambar 5.23 dan detail mengenai fungsinya dijelaskan pada Tabel 5.3.



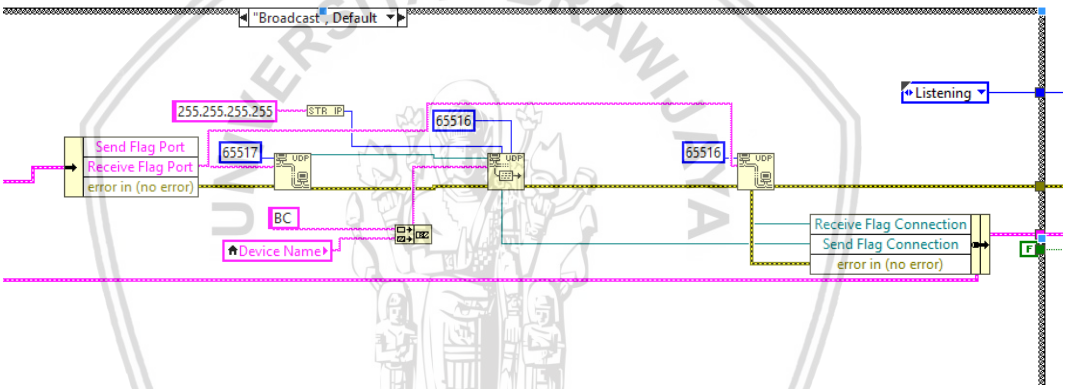
Gambar 5.23 *Inisialisasi constant* yang akan digunakan pada *auto-detect state machine*

Tabel 5.3 Penjelasan mengenai inisialisasi *auto-detect state machine*

Nama Blok	Tipe Data	Fungsi
Send Flag Port	*String*	*Service name* pada blok UDP Open
Receive Flag Port	*String*	*Service name* pada blok UDP Open
Rend Flag Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP untuk mengirimkan *Flag*
Receive Flag Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP untuk menerima *Flag*
Address	*Integer*	Menyimpan alamat IP dari blok UDP
Data Flag	*String*	Menyimpan perintah yang dipakai pada proses *auto detect* perangkat
Identificated Devices	*Array of String*	Menyimpan nama perangkat yang terdeteksi pada jaringan yang sama
Error in	*Error Constant*	Inisialisasi keadaan *error*
Send Ref	*UDP Network*	Menyimpan *Connection ID* yang

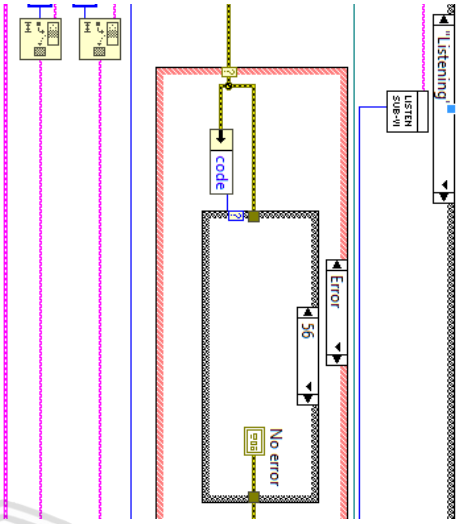
Connection	*Connection*	digunakan oleh semua blok UDP
Device Name	*String*	Menyimpan nama dari perangkat

Pada *state machine* yang pertama dimulai dengan *state Broadcast.* Berdasarkan Gambar 5.24 pada *state* ini dimulai dengan *open port* 65517 dengan *service name* Send Flag. Port ini digunakan untuk mengirim data yang terkait dengan pendeteksian perangkat otomatis seperti BC atau ACK. Kemudian melalui *port* tersebut dilakukan pengiriman data ke 255.255.255.255 port 65516 yang berarti *broadcast* ke semua perangkat pada jaringan yang sama. Data yang dikirim sebagai *broadcast* adalah *string* yang berisi "BC " dan disatukan dengan *Device Name*. Lalu dilakukan *open port* 65516 yang digunakan untuk menerima data *broadcast* maupun ACK. Pembukaan *port* dilakukan setelah pengiriman *broadcast* bertujuan agar data dari *broadcast* tersebut tidak terkirim ke perangkat itu sendiri dan mencegah duplikasi data. *State* selanjutnya adalah *state Listening.*



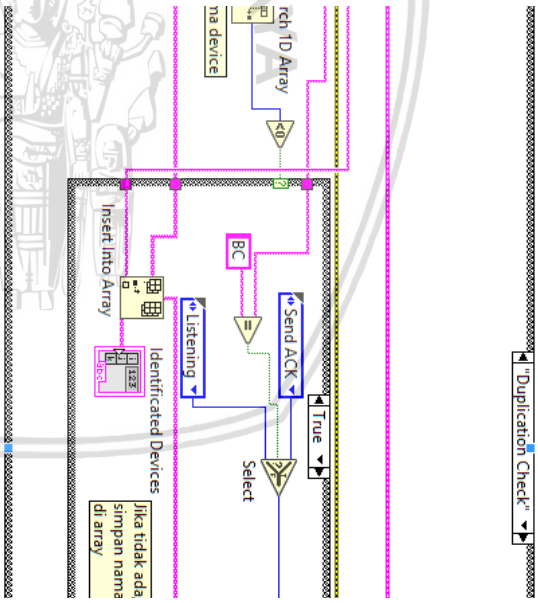
Gambar 5.24 *State Broadcast* pada myRIO

Pada *state Listening* myRIO tidak jauh berbeda dengan *state Listening* yang dimiliki PC seperti pada Gambar 5.9. Saat *Listening* myRIO dapat menerima *broadcast* maupun *ACK* yang dikirimkan oleh perangkat lain. Untuk blok diagram *state Listening* pada myRIO digambarkan pada Gambar 5.25.



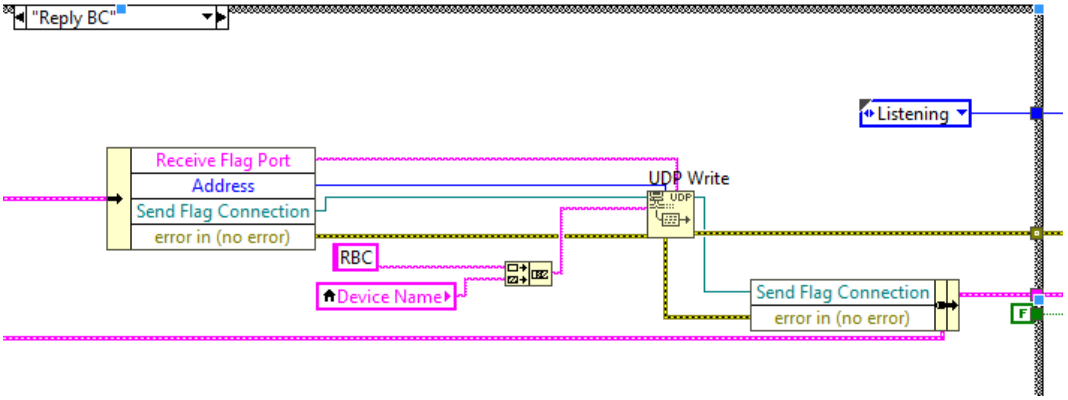
Gambar 5.25 *State Listening pada myRIO*

Pada *state Duplication Check* myRIO juga tidak jauh berbeda dengan *state Duplication Check* dari PC. Namun bedanya pada myRIO tidak membutuhkan pengiriman data "REF" karena myRIO hanya menjadi *client* untuk mengirimkan data status perangkat. Saat *Duplication Check,* myRIO akan menyimpan perangkat yang baru pada *array* dan membiarkan perangkat yang telah dikenali. Blok diagram dari *state Duplication Check* digambarkan pada Gambar 5.26.



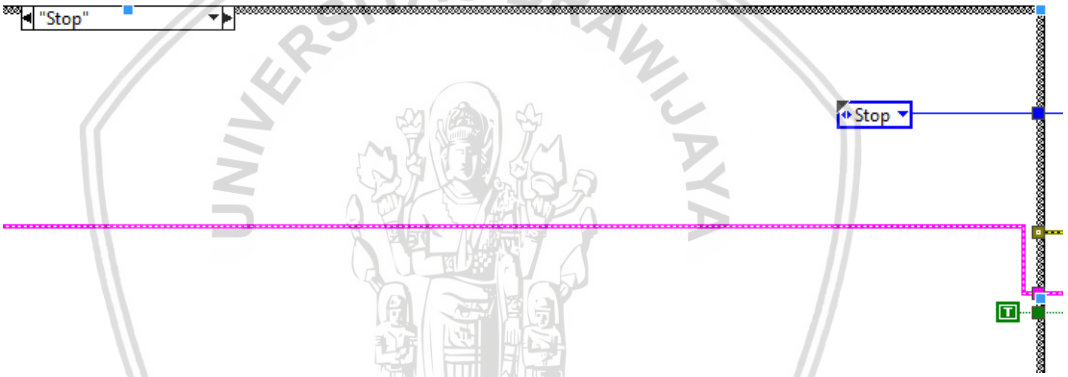
Gambar 5.26 *State Duplication Check* pada myRIO

Pada *state Reply BC* myRIO akan mengirimkan data RBC kepada pengirim *broadcast* agar pengirim dapat mengenali perangkat yang menerima *broadcastnya.* Blok diagram yang digunakan *state Reply BC* diilustrasikan pada Gambar 5.27.



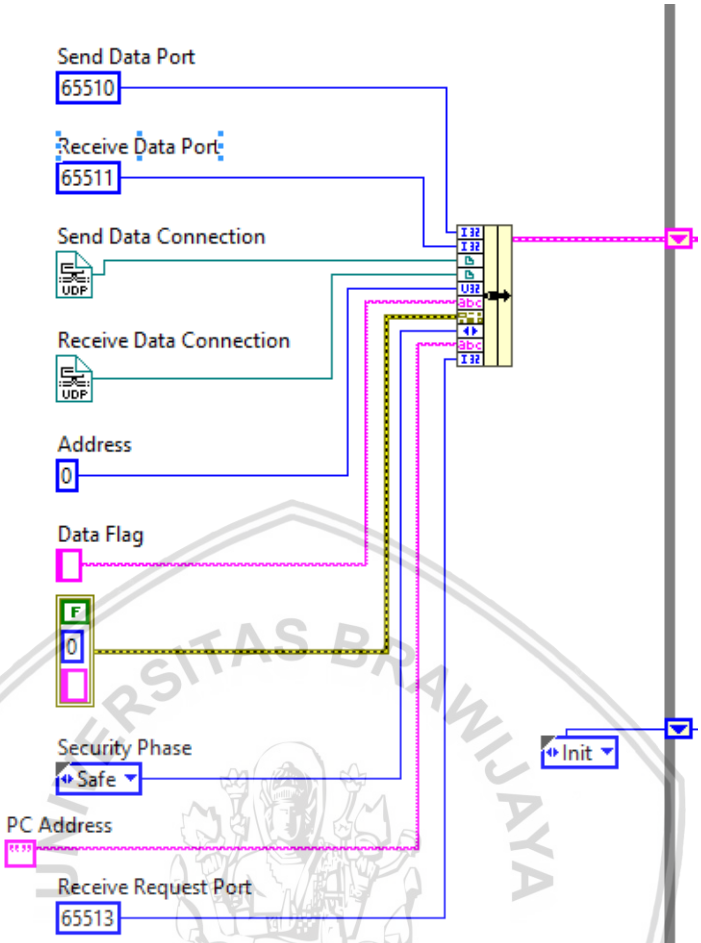
Gambar 5.27 *State Reply BC* pada myRIO

Pada *state Stop,* myRIO akan menghentikan program apabila terdapat penekanan tombol stop pada *front panel*-nya. Dengan mengeluarkan nilai *boolean True* maka *while loop* akan berhenti. Blok diagram yang digunakan diilustrasikan pada Gambar 5.28.



Gambar 5.28 *State Stop* pada myRIO

Untuk *state machine* yang kedua pada myRIO berfungsi untuk proses komunikasi data ke PC. Blok yang digunakan untuk *inisialisasi* tersebut digambarkan pada Gambar 5.29 dan detail mengenai fungsinya dijelaskan pada Tabel 5.4.



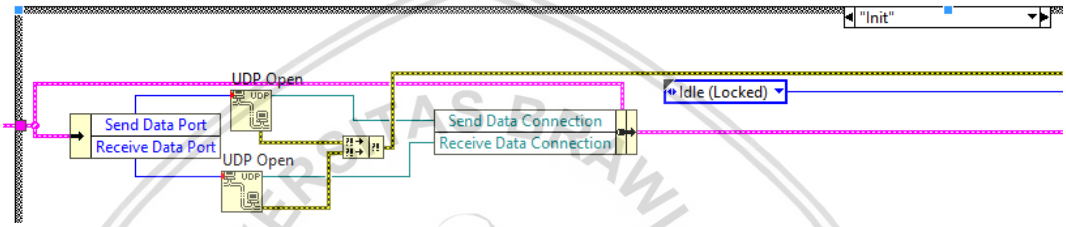
Gambar 5.29 *Inisialisasi state machine* proses pertukaran data pada myRIO

Tabel 5.4 Penjelasan blok yang digunakan *inisialisasi state machine* pertukaran data

Nama Blok	Tipe Data	Fungsi
Send Data Port	*Integer*	*Port* untuk mengirimkan data.
Receive Data Port	*String*	*Port* untuk menerima data.
Rend Data Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP menerima data.
Receive Data Connection	*UDP Network Connection*	Menyimpan *Connection ID* yang digunakan oleh semua blok UDP mengirim data.
Address	*Integer*	Menyimpan alamat IP dari blok UDP.
Data Flag	*String*	Menyimpan perintah yang dipakai pada proses pertukaran data

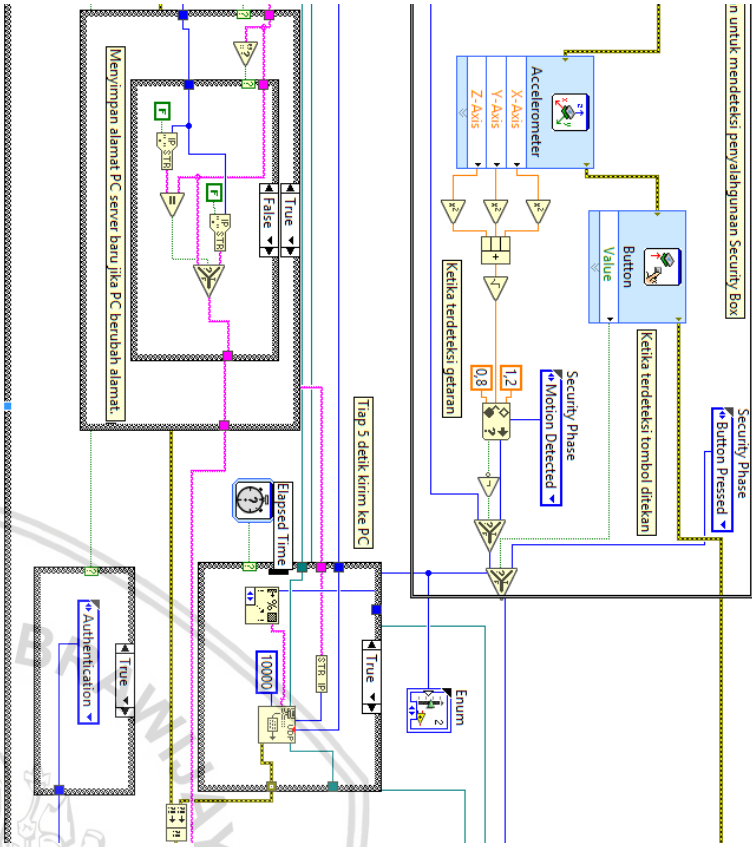
Security Phase	*Enum*	Menyimpan status myRIO yang sedang aktif.
Error in	*Error Constant*	Inisialisasi keadaan *error*
PC Address	*String*	Menyimpan alamat dari PC server.
Receive Request Port	*Integer*	*Port* untuk menerima *request.*

State Init merupakan *state* pertama yang dijalankan pada *state machine* proses pertukaran data pada myRIO. Pada *state* ini dilakukan pembukaan port 65510 dan 65511 untuk mengirim dan menerima data. Setelah itu berganti *state* ke *state Idle (Locked).* Blok diagram yang digunakan digambarkan pada Gambar 5.30.



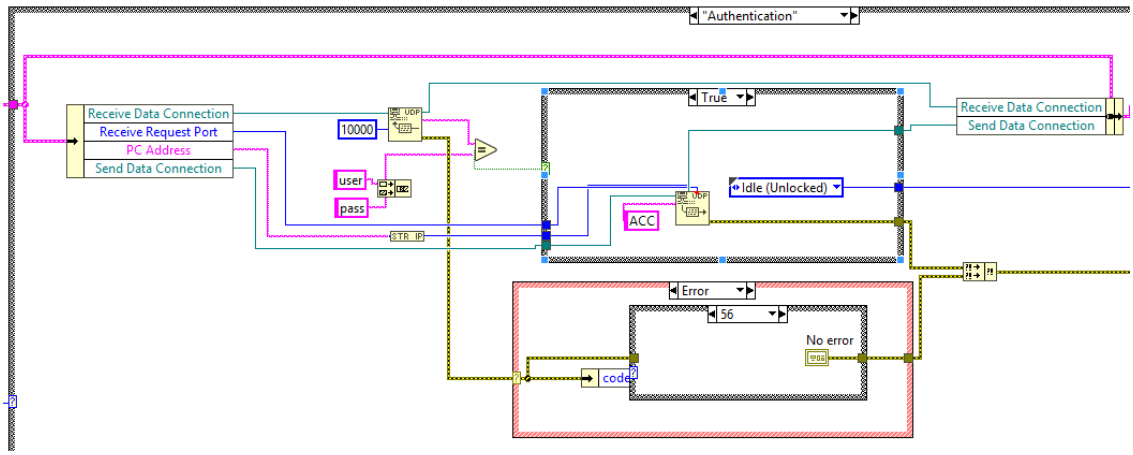
Gambar 5.30 *State Init* pada myRIO

Selanjutnya pada *state Idle (Locked),* myRIO akan mengirimkan status aktif dari perangkat itu kepada PC dalam setiap 5 detik. Status aktif tersebut adalah "*Safe*" berarti myRIO tidak diotak-atik oleh siapapun, "*Motion Detected*" berarti myRIO mendeteksi getaran dari sensor akselerometer, dan "*Button Pressed*" berarti tombol dari myRIO ditekan. Ketika pada *state* ini status telah menjadi "*Button Pressed*" atau "*Motion Detected*" maka tidak bisa kembali menjadi "*Safe*". Pada *state* ini terdapat *UDP Read* untuk menerima data *string* "REQ" untuk melakukan autentikasi perangkat serta berpindah ke *state Authentication* dan data *string* "REF" untuk menyimpan alamat IP dari PC sebagai server. Ketika ada PC dengan alamat baru mengirimkan "REF" ke myRIO, maka myRIO akan mengganti alamat PC yang lama menjadi alamat yang baru didapatkan dari *UDP Read.* Blok diagram yang digunakan pada *state* ini diilustrasikan pada Gambar 5.31.



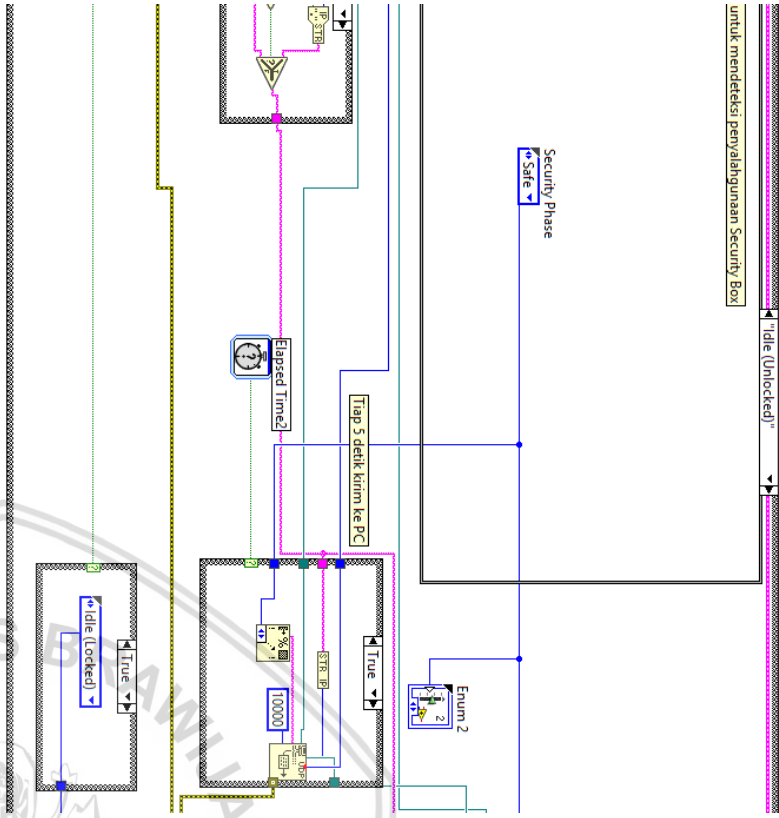
Gambar 5.31 *State Idle (Locked)* pada myRIO

Ketika myRIO mendapatkan data *string "*REQ" pada *state Idle (Locked),* maka myRIO akan menjadi *state Authentication.* Pada *state* ini myRIO akan meminta pengguna PC untuk memasukkan *username* dan *password* agar perangkat myRIO menjadi *state Idle (Unlocked)* dan status perangkat kembali menjadi "*Safe"*. Pertama myRIO akan menunggu data *username* dan *password* yang dikirimkan pada *Receive Request Port* dari myRIO. Setelah data diterima maka data akan dicocokkan menggunakan blok *Equal* dan keluarannya menjadi kondisi dari *Case Structure.* Apabila data yang dimasukkan benar maka myRIO akan mengirimkan string "ACC" kepada *Receive Request Port* dari PC menandakan data benar dan *next state* dari myRIO menjadi *Idle (Unlocked).* Apabila data yang dimasukkan salah maka myRIO mengirimkan *string* "REJ" berarti data salah dan myRIO kembali menjadi *state Idle (Locked).* Blok diagram dari *state Authentication* digambarkan pada Gambar 5.32.



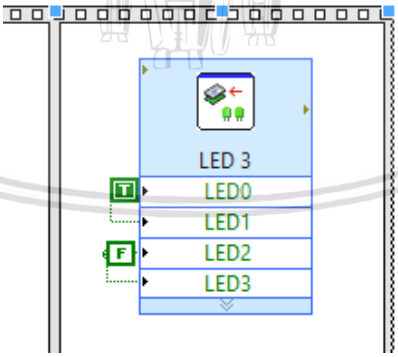
Gambar 5.32 *State Authentication* pada myRIO

Untuk dapat mengembalikan status perangkat menjadi "*Safe*" maka fungsi *Security Box* dari myRIO dinonaktifkan dengan mengakses *state Idle (Unlocked).* Pada state ini perangkat myRIO mengirimkan status "*Safe*" kepada PC setiap 5 detik dan meng-*overwrite* status yang sebelumnya. Pada *state* ini juga PC menampilkan telah terhubung dengan myRIO. Status perangkat akan tetap "*Safe*" meskipun terdapat getaran maupun tombol dari myRIO ditekan. Untuk kembali mengaktifkan fungsi *Security Box* maka myRIO harus kembali ke *state Idle (Locked)* dengan menerima data *string "CLO"* dari PC. Blok diagram dari *state Idle (Unlocked)* digambarkan pada Gambar 5.33.



Gambar 5.33 *State Idle (Unlocked)* pada myRIO

Terakhir ketika *while loop* dari kedua *state machine* berhenti, maka program mengeksekusi *frame* terakhir dengan menyalakan LED dengan pola tertentu menandakan program telah berhenti. Blok yang digunakan pada *frame* terakhir digambarkan pada Gambar 5.34.



Gambar 5.34 *Frame* terakhir dari *Flat Sequence*

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini akan menjelaskan mengenai pengujian kebutuhan fungsional dan kebutuhan non-fungsional dari sistem yang telah diimplementasikan berdasarkan rancangan sebelumnya.

6.1 Pengujian Kebutuhan Fungsional

6.1.1 Pengujian Personal Computer dan myRIO dapat mendeteksi otomatis perangkat lain.

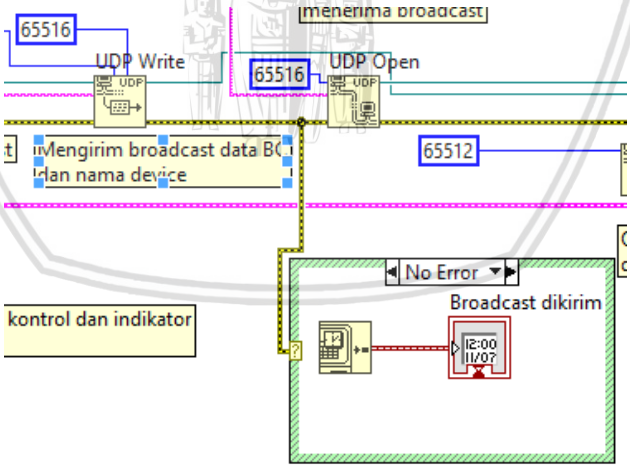
6.1.1.1 Tujuan Pengujian

Pengujian PC dan myRIO dapat mendeteksi perangkat lain secara otomatis adalah untuk melakukan verifikasi serta *discovery time* ketika proses mendeteksi.

6.1.1.2 Prosedur Pengujian

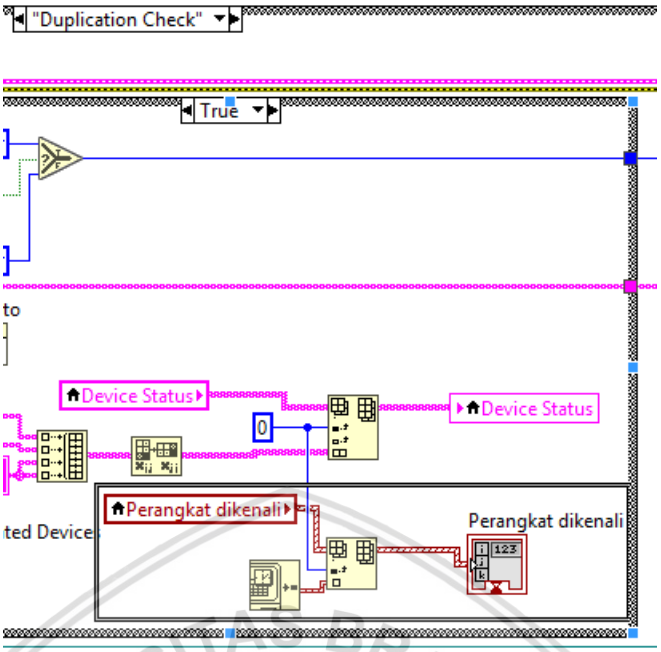
Prosedur yang dilakukan adalah sebagai berikut :

1. Melakukan *state transtition checking* untuk menguji *state machine* pendeteksi perangkat otomatis.
2. Catat waktu mengirim *broadcast* dan waktu perangkat lain dikenali pada PC untuk menghitung *discovery time* ketika terdapat 1 myRIO. Untuk mendapatkan waktu *broadcast* dikirim maka buat indikator untuk menyimpan waktu tersebut. Blok yang digunakan digambarkan pada Gambar 6.1

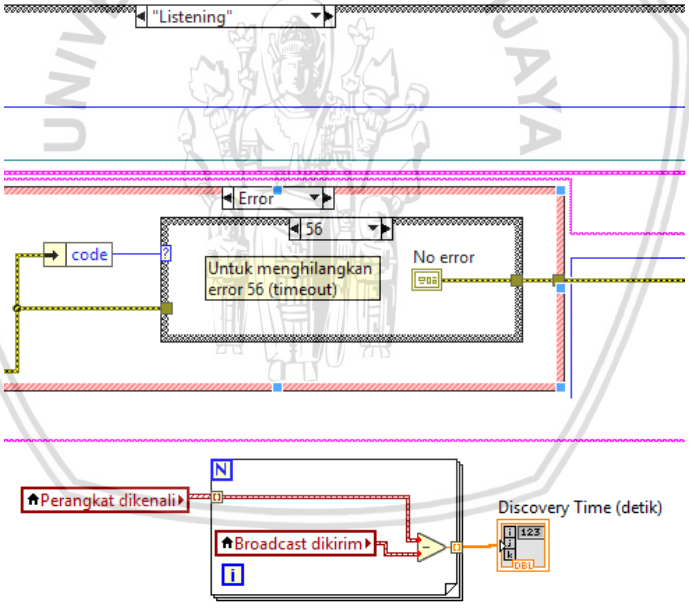


Gambar 6.1 Blok untuk menyimpan waktu *broadcast* dikirim.

Untuk mendapatkan waktu perangkat dikenali dan *discovery time* dibuat indikator *array* untuk menyimpan waktu setiap perangkat dikenali dan selisih waktunya. Blok yang digunakan digambarkan pada Gambar 6.2 dan Gambar 6.3.



Gambar 6.2 Blok untuk menyimpan waktu perangkat dikenali



Gambar 6.3 Blok untuk menghitung selisih waktu (*Discovery Time)*

3. Catat waktu mengirim *broadcast* dan waktu setiap perangkat lain dikenali pada PC untuk menghitung *discovery time* ketika terdapat 2 myRIO.

6.1.1.3 Hasil dan Analisis Pengujian

Untuk dapat menguji mekanisme pendeteksi perangkat otomatis maka dilakukan *state transition checking*. Pengujian ini dilakukan untuk mengecek apakah perpindahan *state* bertindak seperti yang telah dirancang sebelumnya. Hasil dari pengujian ini dijabarkan pada Tabel 6.1.

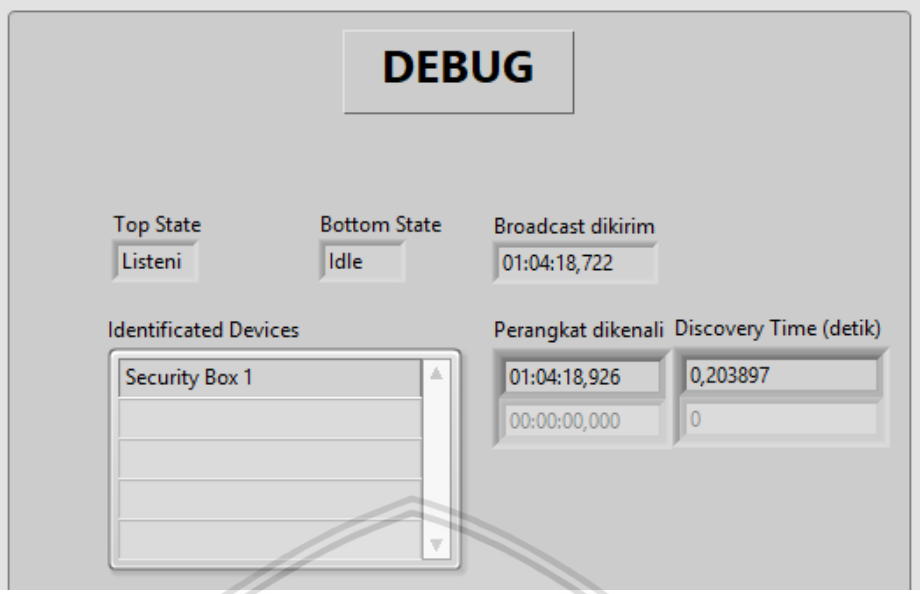
Tabel 6.1 Tabel Pengujian *State Machine* Pendeteksi Perangkat Otomatis

No.	Initial State	Test Case	Expected Destination State and Result	Result	Status
1.	*Broadcast*	Data "BC PC" *dibroadcast* ke 255.255.255.255.	*Listening.*	Setelah *broadcast* dikirim, program memasuki state Listening .	Valid
2.	*Listening*	Menerima data flag "ACK" dari perangkat lain.	*Check Duplication.*	Setelah menerima "ACK", program memasuki state Check Duplication.	Valid
3.		Menerima data flag "BC " dari perangkat lain.	*Check Duplication*	Setelah menerima "BC ", program memasuki *state Check Duplication.*	Valid
4.		Pengguna menekan tombol Stop	Stop	Program berpindah ke *state Stop.*	Valid
5.	Check Duplication	Data flag sebelumnya "BC " dan perangkat lain belum terdapat pada array.	Simpan ke array. Reply BC.	Nama dan alamat perangkat baru tersimpan pada *array*. Program berpindah ke *state Reply BC.*	Valid
6.		Data flag sebelumnya "BC " dan perangkat	Reply BC.	Program berpindah ke *state Reply*	Valid

		lain telah tersimpan pada array.		*BC*	
7.		Data flag sebelumnya "ACK" dan perangkat lain belum terdapat pada array	Simpan ke array. Listening	Nama dan alamat perangkat baru tersimpan pada *array.* Program berpindah ke *state Reply BC*	Valid
8.		Data flag sebelumnya "ACK" dan perangkat lain telah tersimpan pada array.	Listening.	Program berpindah ke *state Listening*	Valid
9.	Reply BC	Mengirim data "RBC" kepada pengirim *broadcast.*	Listening.	Setelah mengirim "RBC", program kembali ke *state Listening*	Valid
10.	Stop	Program menghentikan while loop.	Stop	Program berhenti karena while loop telah mendapat kondisi true untuk berhenti.	Valid

Berdasarkan Tabel 6.1 dapat disimpulkan bahwa pengujian menggunakan *state transtition checking* dinyatakan 100% berhasil diuji sesuai dengan rancangan berdasarkan status yang didapatkan setiap pengujian.

Untuk mendapatkan *Discovery Time* dibutuhkan selisih waktu antara *broadcast* dikirimkan hingga perangkat lain dikenali oleh PC. Hasil dari pengujian *discovery time* ketika terdapat 1 myRIO yang sedang aktif diilustrasikan pada Gambar 6.4.

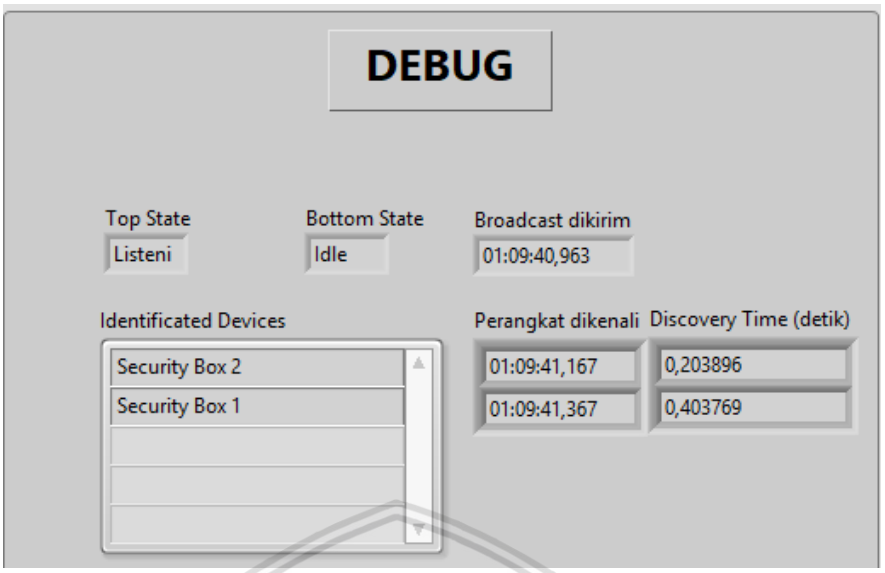


Gambar 6.4 Pengujian *discovery time* untuk 1 myRIO

Berdasarkan Gambar 6.4 didapatkan *discovery time* selama 0,203897 detik sehingga rentang waktu perangkat dikenali relatif cepat. Dilakukan pengujian *discovery time* sebanyak 5 kali dan didapatkan hasil seperti pada Tabel 6.2. Berdasarkan Tabel 6.2 didapatkan nilai rata-rata *discovery time* selama 0,202754 detik. Sedangkan hasil pengujian *discovery time* ketika ada 2 myRIO yang sedang aktif digambarkan pada Gambar 6.5.

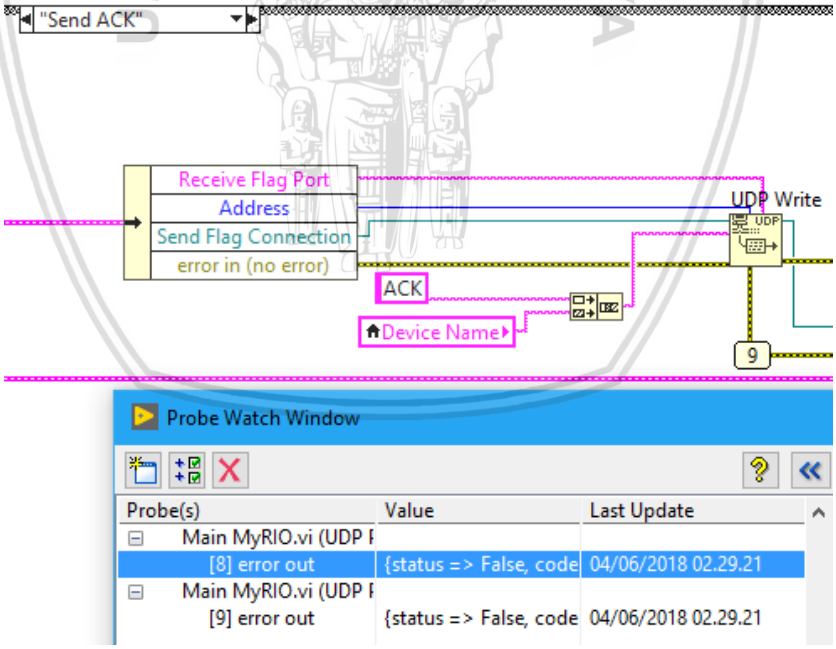
Tabel 6.2 Pengujian *discovery time* untuk 1 myRIO

No.	Broadcast Dikirim (waktu)	Perangkat Dikenali (waktu)	Discovery Time (detik)
1.	03:33:32,244	03:33:32,448	0,20434
2.	03:33:50,539	03:33:50,741	0,201762
3.	03:34:14,432	03:34:14,634	0,201895
4.	03:34:43,418	03:34:43,620	0,201886
5.	03:34:59,555	03:34:59,758	0,203888
	Rata-rata		165,973



Gambar 6.5 Pengujian *discovery time* untuk 2 myRIO

Berdasarkan Gambar 6.5 didapatkan *discovery time* selama 0,203896 detik untuk myRIO pertama dan 0,403769 detik untuk myRIO kedua. Kedua perangkat myRIO mengirimkan ACK secara bersamaan kepada PC berdasarkan Gambar 6.6. Pada Gambar 6.6 terlihat bahwa kedua myRIO berhasil mengirimkan RBC secara bersamaan pada waktu 2.29.31 kepada PC.



Gambar 6.6 Waktu pengiriman RBC dari kedua myRIO

Dilakukan pengujian *discovery time* sebanyak 5 kali dan didapatkan data seperti pada Tabel

Tabel 6.3 Pengujian *discovery time* untuk 2 myRIO

No.	Broadcast Dikirim (waktu)	Perangkat Dikenali (waktu)	Discovery Time (detik)
1.	03:49:28,302	03:49:28,505	0,203223
		03:49:28,706	0,404165
2.	03:50:04,304	03:50:04,507	0,202888
		03:50:04,708	0,403768
3.	03:50:25,899	03:50:26,101	0,201886
		03:50:26,301	0,401786
4.	03:50:45,630	03:50:45,834	0,203907
		03:50:46,035	0,405414
5.	03:52:21,931	03:52:22,133	0,202542
		03:52:22,333	0,402426
	Rata-rata :		303,200.5

Setiap baris pada Tabel 6.3 kolom Perangkat Dikenali dan Discovery Time menandakan hasil pada perangkat myRIO pertama dan perangkat myRIO kedua dari waktu Broadcast Dikirim yang sama. Berdasarkan Tabel 6.3 didapatkan rata-rata kedua perangkat myRIO dikenali dalam waktu 0,3 detik dan lebih lama 0,1 detik dibandingkan dengan *discovery time* ketika hanya ada 1 myRIO. Serta myRIO kedua dikenali sekitar 0,2 detik setelah myRIO pertama dikenali. Berdasarkan hasil yang didapatkan dapat disimpulkan bahwa ketika kedua myRIO mengirimkan ACK secara bersamaan, PC hanya dapat memproses satu data ACK tersebut dalam satu waktu. PC tidak dapat menyimpan kedua myRIO secara bersamaan dan hanya dapat memprosesnya secara sekuensial.

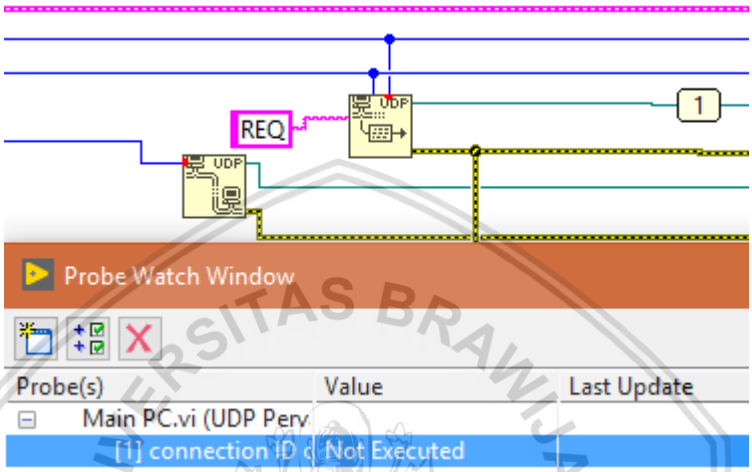
6.1.2 Pengujian Perangkat myRIO dapat bertindak sebagai Security Box.

6.1.2.1 Tujuan Pengujian

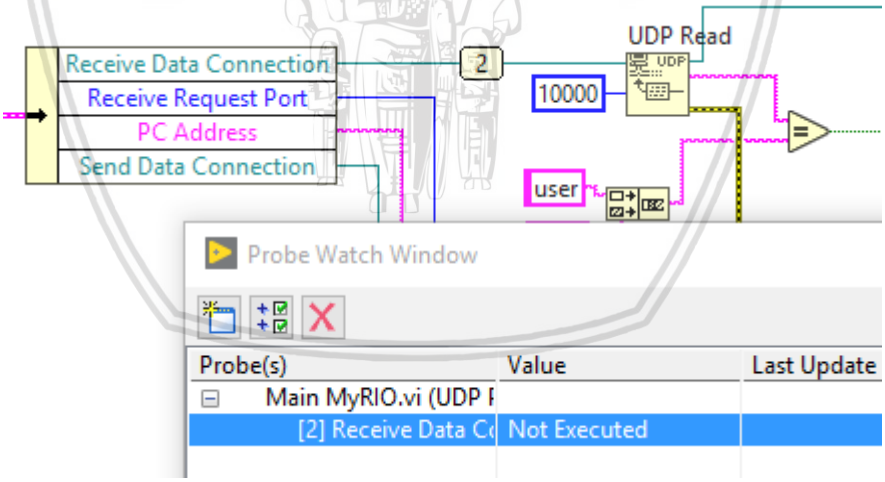
Pengujian perangkat myRIO dapat bertindak sebagai *Security Box* bertujuan untuk mengetahui validitas implementasi *state machine* pertukaran data pada PC dengan myRIO serta untuk mengetahui *delay* yang terjadi saat pengiriman data.

6.1.2.2 Prosedur Pengujian

1. Melakukan *state transition checking* untuk menguji *state machine* proses pertukaran data.
2. Catat waktu pengiriman *request* dari PC dan waktu myRIO berpindah ke *state Authentication* untuk menghitung *delay* ketika meminta autentikasi. Untuk mendapatkan waktu tersebut menggunakan fungsi *Probe* yang diletakkan pada *wire* keluaran dari pengiriman request seperti pada Gambar 6.7 dan pada *wire* masukan dari blok UDP Read seperti pada Gambar 6.8.

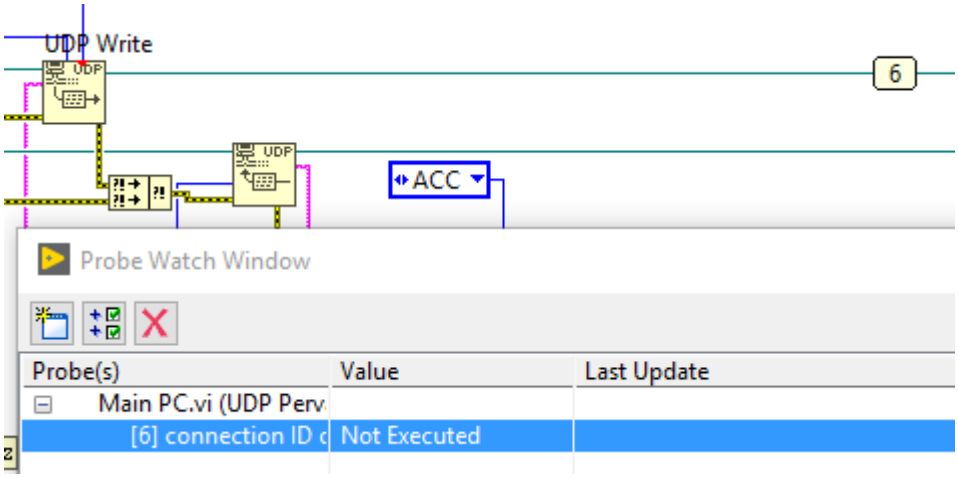


Gambar 6.7 *Probe* untuk mendapat waktu mengirim request.

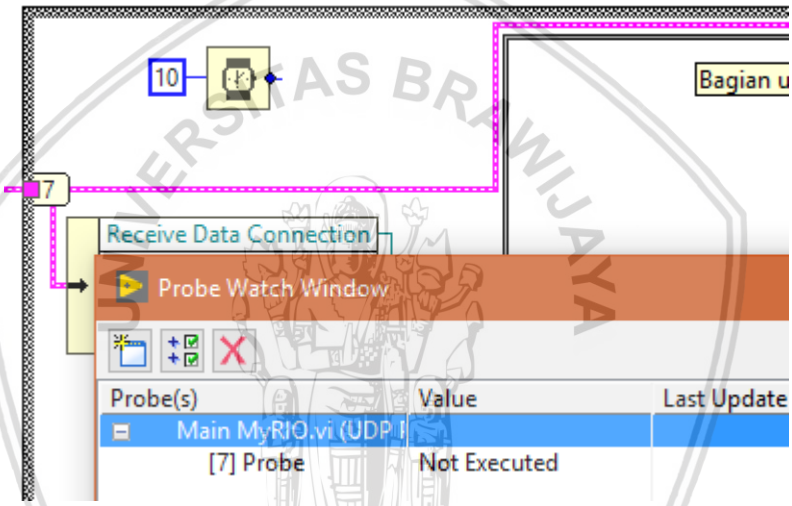


Gambar 6.8 *Probe* untuk mendapat waktu myRIO menjadi mode Autentikasi

Catat waktu pengiriman *username password* dan waktu myRIO berpindah ke *state Idle (Unlocked)* untuk menghitung delay ketika proses *unlocking* menggunakan *Probe*. *Probe* digunakan pada *wire* keluaran dari blok UDP Write untuk mengirim *username password* seperti pada Gambar 6.9 dan pada *wire* dari *cluster* seperti pada Gambar 6.10.

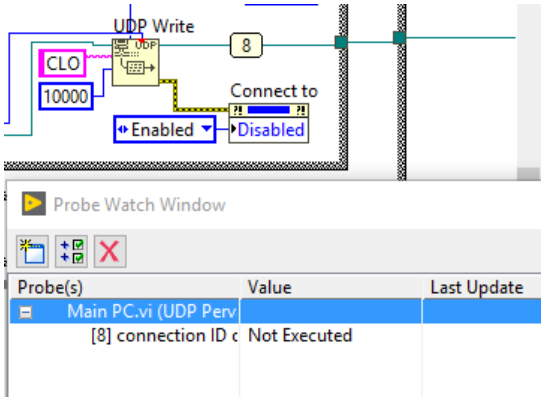


Gambar 6.9 *Probe* untuk mendapat waktu pengiriman *username password*

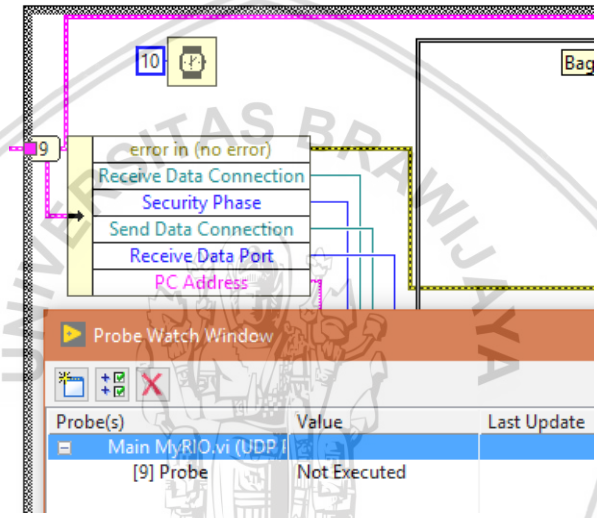


Gambar 6.10 *Probe* untuk mendapat waktu myRIO berpindah menjadi *Unlocked*

3. Catat waktu pengiriman penguncian myRIO dan waktu myRIO kembali ke *state Idle (Locked)* untuk menghitung delay ketika proses *locking* menggunakan *Probe. Probe* digunakan pada *wire* keluaran dari blok UDP Write seperti pada Gambar 6.11 dan pada *wire* dari cluster seperti pada Gambar 6.12.



Gambar 6.11 *Probe* yang digunakan untuk mendapat waktu pengiriman penguncian myRIO



Gambar 6.12 *Probe* yang digunakan untuk mendapat waktu myRIO kembali menjadi Locked

6.1.2.3 Hasil dan Analisis Pengujian

Untuk pengujian *state machine* pertukaran data maka dilakukan pengujian *state transition checking* pada myRIO dan pada PC sebab *state machine* pada kedua perangkat tersebut berbeda. Hasil dari pengujian *state transition checking* myRIO dijabarkan pada Tabel 6.4 dan hasil dari pengujian *state transition checking* PC dijabarkan pada Tabel 6.5.

Tabel 6.4 Pengujian *state machine* pertukaran data pada myRIO

No.	Initial State	Test Case	Expected Destination State and Result	Result	Status
1.	*Init*	Pembukaan port untuk mengirim dan menerima data.	*Idle (Locked).* Status *Safe*	Setelah *port* dibuka, program memasuki state *Idle (Locked)* dan status perangkat *Safe.*	Valid
2.	*Idle (Locked).* Status *Safe*	Mendeteksi adanya getaran pada myRIO.	*Idle (Locked).* Status *Motion Detected*	Setelah getaran terdeteksi, status perangkat berubah menjadi *Motion Detected* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	Valid
3.		Tombol pada myRIO ditekan.	*Idle (Locked).* Status *Button Pressed*	Setelah tombol ditekan, status perangkat berubah menjadi *Button Pressed* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke	Valid

				PC dengan jeda 5 detik.	
4.		PC mengirimkan *request* untuk autentikasi perangkat.	*Authentication.*	Program berpindah ke *state Authentication.*	Valid
5.	*Idle (Locked).* Status *Motion Detected.*	Mendeteksi adanya getaran pada myRIO.	*Idle (Locked).* Status *Motion Detected*	Setelah getaran terdeteksi, status perangkat tetap *Motion Detected* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	Valid
6.		Tombol pada myRIO ditekan.	*Idle (Locked).* Status *Button Pressed*	Setelah tombol ditekan, status perangkat berubah menjadi *Button Pressed* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	Valid
7.		PC mengirimkan *request* untuk autentikasi perangkat.	*Authentication.*	Program berpindah ke *state Authentication.*	Valid
8.	*Idle (Locked).* Status *Button Pressed.*	Mendeteksi adanya getaran pada	*Idle (Locked).* Status *Motion Detected*	Setelah getaran terdeteksi,	Valid

		myRIO.		status perangkat berubah menjadi *Motion Detected* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	
9.		Tombol pada myRIO ditekan.	*Idle (Locked).* Status *Button Pressed*	Setelah tombol ditekan, status perangkat tetap *Button Pressed* dan *state* masih *Idle (Locked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	Valid
10		PC mengirimkan *request* untuk autentikasi perangkat.	*Authentication.*	Program berpindah ke *state Authentication.*	
11.	*Authentication.*	Pengguna PC memasukkan *username* dan *password* dengan benar dalam jangka waktu di bawah 10 detik.	*State Idle (Unlocked). Status Safe.*	myRIO mengirimkan data ACC kepada PC. Program berpindah ke *state Idle (Unlocked)* dan status perangkat menjadi *Safe.*	Valid
12.		Pengguna PC memasukkan *username* dan	*State Idle (Locked).*	myRIO mengirimkan data REJ	Valid

		password dengan salah dalam jangka waktu di bawah 10 detik.		kepada PC. Program berpindah ke *state Idle (Locked)* dan status perangkat tetap.	
13.		Pengguna tidak memasukkan *username* dan *password* selama 10 detik.	*State Idle (Locked).*	myRIO mengirimkan data REJ kepada PC. Program berpindah ke *state Idle (Locked)* dan status perangkat tetap.	Valid
14.	*Idle (Unlocked).* Status *Safe*.	Mendeteksi adanya getaran pada myRIO.	*Idle (Unlocked).* Status *Safe.*	Setelah getaran terdeteksi, status perangkat tetap *Safe* dan *state* masih *Idle (Unlocked).* Status perangkat dikirimkan ke PC dengan jeda 5 detik.	Valid
15.		Tombol pada myRIO ditekan.	*Idle (Unlocked).* Status *Safe.*	Setelah tombol ditekan, status perangkat tetap *Safe* dan *state* masih *Idle (Unlocked).* Status perangkat dikirimkan ke PC dengan jeda	Valid

				5 detik.	
16.		PC mengirimkan CLO untuk mengunci perangkat.	*Idle (Locked).* Status *Safe*	Program berpindah ke *state Idle (Locked) dan* status tetap *Safe.*	Valid

Berdasarkan hasil yang didapatkan pada Tabel 6.4, dapat disimpulkan bahwa *state machine* pertukaran data pada myRIO dinyatakan 100% berhasil diuji sesuai dengan rancangan.

Tabel 6.5 Pengujian *state machine* pertukaran data pada PC

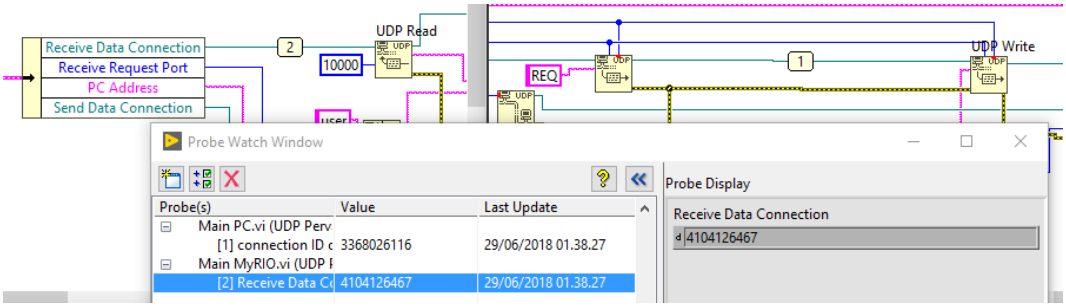
No.	Initial State	Test Case	Expected Destination State and Result	Result	Status
1.	*Init*	Pembukaan port untuk mengirim dan menerima data.	*Listening*	Setelah *port* dibuka, program memasuki state *Idle (Locked)* dan status perangkat *Safe.*	Valid
2.	*Listening.* Tombol Connect nonaktif.	Menerima data status perangkat *(Safe, Motion Detected, Button Pressed)* secara rutin di bawah 10 detik	*State Listening. Device Status* menampilkan status perangkat yang dikirimkan myRIO.	Setelah status perangkat diterima. Pada *array Device Status* menampilkan status perangkat yang dikirimkan dan waktu terakhir diterima.	Valid
3.		Tidak menerima data status perangkat *(Safe, Motion*	*State Listening. Device Status* menampilkan *Device Lost* pada	Status perangkat pada *Device Status* berubah	Valid

		Detected, Button Pressed) secara rutin di bawah 10 detik	perangkat tersebut.	menjadi *Device Lost. State* masih Listening.	
4.		Pengguna mengaktifkan tombol *Connect*.	*State Authentication* .	Program berpindah ke *state Authenticatio n* dan membawa alamat perangkat yang akan dihubungkan.	Valid
5.	Authentication .	Pengguna memasukkan *username* dan *password* perangkat dengan benar dalam jangka waktu 10 detik.	State Listening dengan menampilkan perangkat telah terhubung.	PC menerima data ACC yang berarti data yang dimasukkan benar dan kembali ke *state Listening* dengan tombol *Connect* yang masih aktif dan menampilkan perangkat yang dihubungkan.	Valid
6.		Pengguna memasukkan *username* dan *password* perangkat dengan salah dalam jangka waktu 10 detik.	State Listening dengan menampilkan data yang dimasukkan salah.	PC menerima data REJ yang berarti data yang dimasukkan salah dan kembali ke *state Listening* dengan tombol *Connect* yang	

				kembali nonaktif.	
7.		Pengguna tidak memasukkan *username* dan *password* perangkat dalam jangka waktu 10 detik.	State Listening dengan menampilkan komunikasi gagal.	PC menerima data REJ dan kembali ke *state Listening* dengan tombol *Connect* yang kembali nonaktif.	Valid
9.	*State Listening.* Tombol Connect Aktif	Pengguna menonaktifkan tombol *Connect.*	State Listening dengan menampilkan perangkat telah *locked*.	PC mengirimkan data "CLO" untuk mengunci myRIO. myRIO kembali pada state Idle (Locked) dan pada PC tombol *Connect* menjadi nonaktif.	Valid

Berdasarkan hasil pengujian yang didapatkan pada Tabel 6.5, dapat disimpulkan bahwa *state machine* pertukaran data pada PC telah berhasil diuji 100% dan sesuai dengan rancangan yang telah dibuat.

Untuk menguji *delay* akan menggunakan fitur *Probe* lalu mendapatkan waktu dari *last update probe* tersebut tereksekusi. *Probe* digunakan karena pada myRIO sendiri tidak memiliki fungsi RTC sehingga waktu pada PC tidak sama dengan waktu pada myRIO. Probe akan menggunakan waktu pada PC meskipun dieksekusi pada perangkat myRIO. Namun kelemahan *Probe* ini adalah tingkat ketelitian waktu yang rendah karena tidak dapat mengetahui milidetik dari waktu.

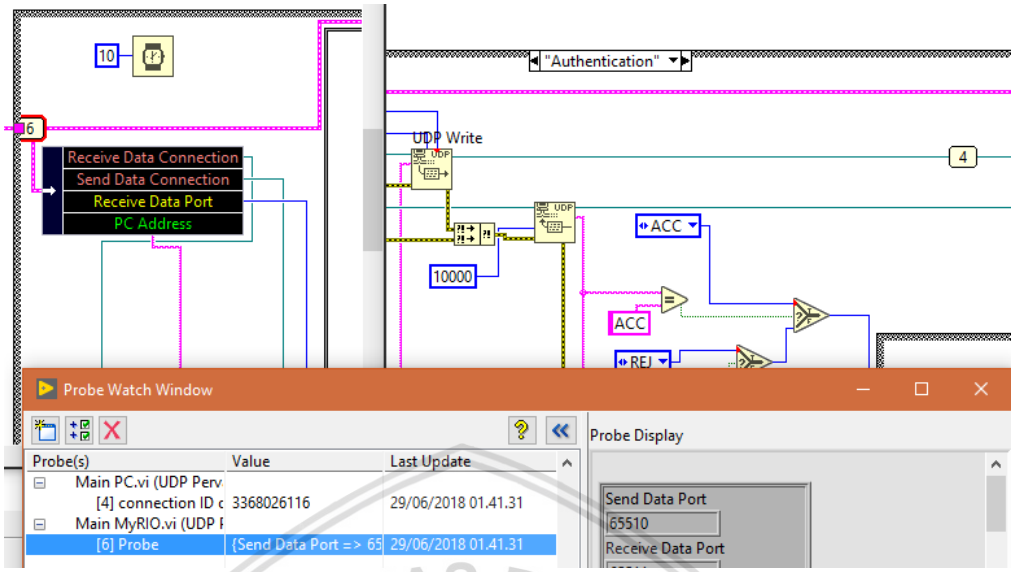


Gambar 6.13 Hasil pengujian *delay* pengiriman *request*

Berdasarkan Gambar 6.13 *delay* pengiriman *request* dari PC kepada myRIO selama kurang dari 1 detik yang didapatkan dari selisih waktu *probe* 1 dengan *probe* 2. Dilakukan 5 kali pengujian *delay* pengiriman *request* dan didapatkan hasil seperti pada Tabel 6.6. Berdasarkan Tabel 6.6 didapatkan rata-rata *delay* pengiriman *request* selama 0 detik dan dapat disimpulkan bahwa *delay* pengiriman *request* berlangsung relatif cepat.

Tabel 6.6 Hasil pengujian *delay* pengiriman *request*

No.	Waktu Pengiriman (waktu)	Waktu Diterima (waktu)	Selisih Waktu (detik)
1.	03.02.02	03.02.02	0
2.	03.02.12	03.02.12	0
3.	03.02.24	03.02.24	0
4.	03.02.31	03.02.31	0
5.	03.02.54	03.02.54	0

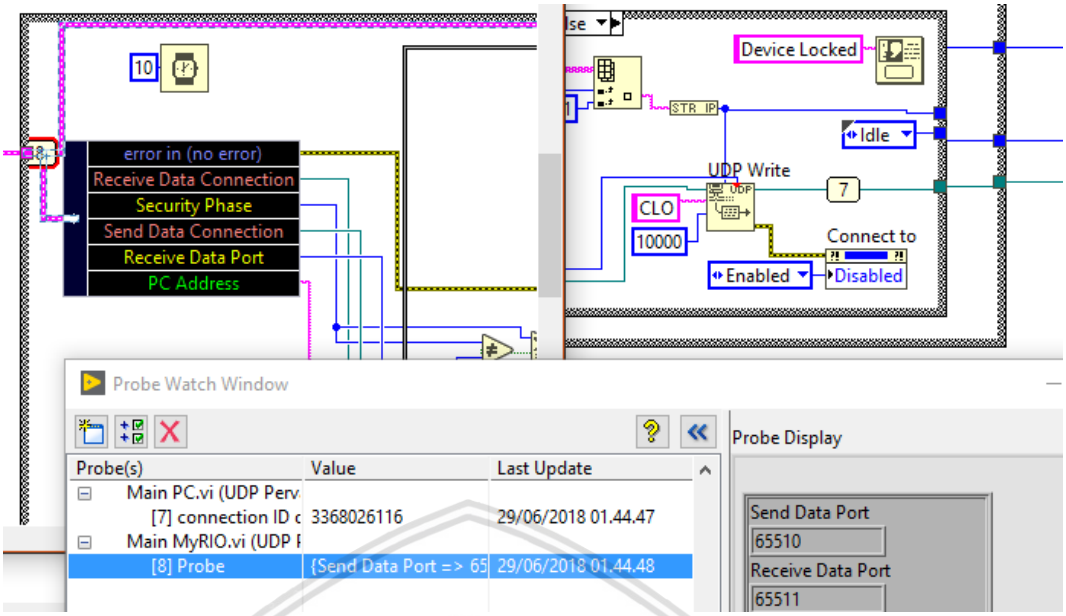


Gambar 6.14 Hasil pengujian *delay* perubahan status menjadi *Unlocked*

Berdasarkan Gambar 6.14 *delay* perubahan status myRIO menjadi *Unlocked* selama kurang dari 1 detik yang didapatkan dari selisih *probe* nomor 4 dengan *probe* nomor 6. Dilakukan 5 kali pengujian *delay* perubahan status myRIO menjadi *Unlocked* dan didapatkan hasil seperti pada Tabel 6.7. Berdasarkan Tabel 6.7 rata-rata *delay* dari perubahan status menjadi *Unlocked* selama 0 detik. Dengan ini dapat disimpulkan bahwa proses perubakan status myRIO menjadi *Unlocked* berlangsung relatif cepat.

Tabel 6.7 Hasil pengujian *delay* perubahan status menjadi *Unlocked*

No.	Waktu Pengiriman (waktu)	Waktu Diterima (waktu)	Selisih Waktu (detik)
1.	03.16.51	03.16.51	0
2.	03.17.46	03.17.46	0
3.	03.18.10	03.18.10	0
4.	03.18.39	03.18.39	0
5.	03.19.15	03.19.15	0



Gambar 6.15 Hasil pengujian *delay* perubahan status menjadi *Locked*

Berdasarkan Gambar 6.15 *delay* perubahan status myRIO menjadi *Locked* selama 1 detik yang didapatkan dari selisih *probe* nomor 7 dengan *probe* nomor 8. Dilakukan 5 kali pengujian *delay* perubahan status myRIO menjadi *Locked* dan didapatkan hasil seperti pada Tabel 6.8. Berdasarkan Tabel 6.8 rata-rata *delay* dari perubahan status menjadi *Locked* selama 0,6 detik. Dengan ini dapat disimpulkan bahwa proses perubahan status myRIO menjadi *Locked* berlangsung relatif lebih lambat dibandingkan dengan proses sebelumnya.

Tabel 6.8 Hasil pengujian *delay* perubahan status menjadi *Locked*

No.	Waktu Pengiriman (waktu)	Waktu Diterima (waktu)	Selisih Waktu (detik)
1.	03.21.52	03.21.52	0
2.	03.22.54	03.22.55	1
3.	03.23.35	03.23.36	1
4.	03.24.19	03.24.20	1
5.	03.15.33	03.25.33	0

6.2 Pengujian Kebutuhan Non-Fungsional

6.2.1 Tujuan Pengujian

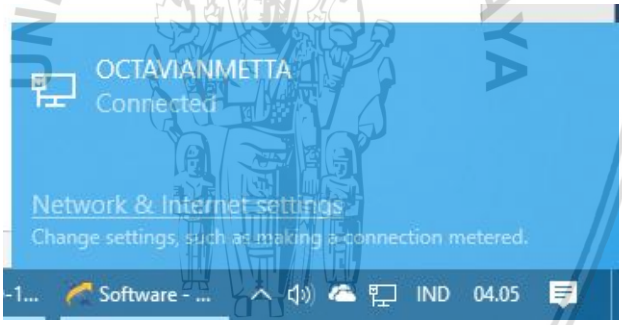
Pengujian kebutuhan non-fungsional ditujukan untuk memastikan bahwa kebutuhan non-fungsional yang dianalisis sebelumnya dapat mendukung dalam mengimplementasikan sistem ini.

6.2.2 Prosedur Pengujian

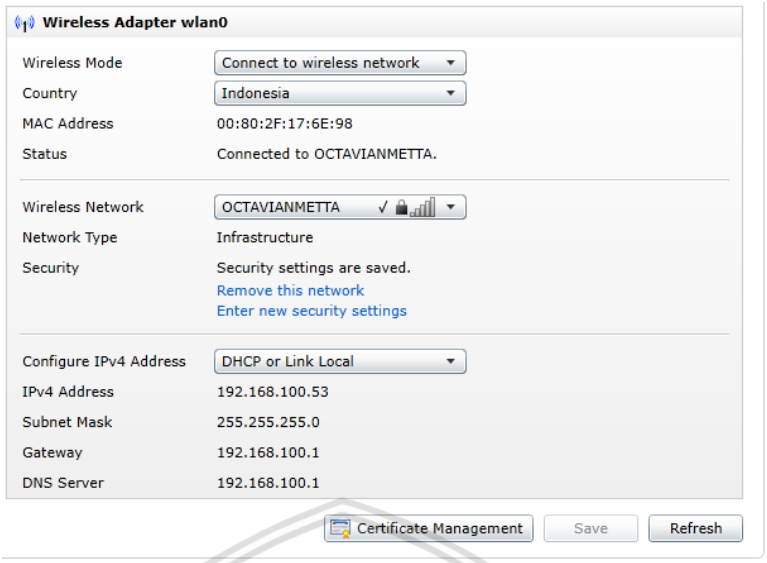
1. Pengecekan perangkat myRIO dan PC terhubung pada *access point* yang sama.
2. Lakukan ping alamat myRIO dari PC. Alamat IP dari myRIO didapatkan pada pengaturan WiFi dari myRIO.

6.2.3 Hasil dan Analisis Pengujian

Untuk menguji PC dan myRIO berada pada jaringan yang sama maka dilakukan pengecekan nama jaringan yang terhubung pada kedua perangkat tersebut. Berdasarkan Gambar 6.16 dan Gambar 6.17 dapat disimpulkan bahwa kedua perangkat tersebut terhubung pada jaringan yang sama.



Gambar 6.16 PC yang telah terhubung pada jaringan



Gambar 6.17 myRIO yang telah terhubung pada jaringan

Hal ini juga diperkuat dengan PC sukses melakukan *ping* kepada alamat dari perangkat myRIO seperti yang diilustrasikan pada Gambar 6.18.

```
C:\Users\Oktavian>ping 192.168.100.53

Pinging 192.168.100.53 with 32 bytes of data:
Reply from 192.168.100.53: bytes=32 time=3ms TTL=64
Reply from 192.168.100.53: bytes=32 time=2ms TTL=64
Reply from 192.168.100.53: bytes=32 time=2ms TTL=64
Reply from 192.168.100.53: bytes=32 time=2ms TTL=64

Ping statistics for 192.168.100.53:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 3ms, Average = 2ms
```

Gambar 6.18 Hasil ping dari PC kepada myRIO

BAB 7 PENUTUP

Bab ini menjelaskan mengenai kesimpulan yang didapatkan selama penelitian ini, dan saran untuk pengembangan selanjutnya dari penelitian ini.

7.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis dari rancangan dan implementasi yang telah dibuat dapat diambil kesimpulan sebagai berikut :

1. Protokol UDP *Pervasive Multi Device* dapat menggunakan desain sistem *state machine* ganda yang memiliki fungsi tertentu. *State machine* pertama digunakan untuk mendeteksi alamat perangkat lain secara otomatis. Sedangkan *state machine* kedua digunakan sebagai aplikasi dari penggunaan alamat perangkat lain yang telah didapatkan.
2. Protokol UDP *Pervasive Multi Device* dapat diimplementasikan pada platform PC maupun myRIO serta dapat mengenali seluruh perangkat dengan menggunakan program berbasiskan LabVIEW. Program dibuat dengan mengimplementasikan desain sistem *state machine* ganda. Pada *state machine* pertama pada PC dan pada myRIO hampir identik dan berfungsi untuk mendeteksi perangkat. Pada *state machine* kedua pada PC digunakan sebagai *host* dari sistem *Security Box.* Sedangkan *state machine* kedua pada myRIO berfungsi sebagai *client* dari sistem *Security Box.*
3. *Security Box* yang merupakan salah satu aplikasi dari perangkat *embedded* berhasil diuji menggunakan protokol UDP *Pervasive Multi Device*. Dalam pengujian *delay* perangkat dikenali serta *delay* komunikasi antar perangkat dinilai responsif. Didapatkan *discovery time* rata-rata 0,202754 detik ketika terdapat 1 myRIO dan *discovery time* rata-rata 0,303201 detik ketika terdapat 2 myRIO. *Delay* pengiriman data dari PC kepada myRIO tidak lebih dari 2 detik. State machine yang digunakan telah diuji menggunakan *state based testing* dan 100% sesuai dengan rancangan.

7.2 Saran

Terdapat beberapa saran yang berguna untuk pengembangan penelitian ini ke depannya sebagai berikut :

1. Desain sistem dua *state machine* dalam satu sistem *loop* dapat menjadi dasar dari pengembangan aplikasi protokol UDP *Pervasive Multi Device* berbasis LabVIEW ke depannya. *State machine* pertama tetap untuk mengenali perangkat menggunakan UDP *Pervasive*. Sedangkan *state machine* kedua dapat untuk aplikasi lain dari implementasi UDP *Pervasive*.

DAFTAR PUSTAKA

Dourish, P., & Bell, G. (2011). *Divining a digital future: Mess and mythology in ubiquitous computing.* MIT Press.

Halvorsen, H.-P. (2016, September 7). Dipetik Januari 15, 2018, dari http://home.hit.no/~hansha/documents/labview/training/Introduction%20to%20LabVIEW/Introduction%20to%20LabVIEW.pdf

Hashemi, S. H., Faghri, F., Rausch, P., & Campbell, R. H. (2016). World of Empowered IoT Users. *IEEE First International Conference on Internet-of-Things Design and Implementation*, 13-24.

Kurniawan, W., Ichsan, M. H., & Akbar, S. R. (2017). UDP Pervasive Protocol Implementation for Smart Home Environment on MyRIO using LabVIEW. *International Journal of Electrical and Computer Engineering (IJECE)*, 113-123.

Kurniawan, W., Ichsan, M., Akbar, S., & Arwani, I. (2017). Lightweight UDP Pervasive Protocol in Smart Home. *IOP Conference Series: Materials Science and Engineering, Volume 190, conference 1* , 012009.

Kurose, J. F., & Ross, K. W. (2010). *Computer Networking: A Top-Down Approach (5th ed.).* Boston: Pearson Education.

Lamey, D. (2018, Januari 5). *Past, Present and Future: The Evolution of Technology*. Dipetik Januari 10, 2018, dari DiscoverTec Web site: http://www.discovertec.com/blog/the-evolution-of-technology

National Instruments. (2014, April 1). *From Student to Engineer: Preparing Future Innovators With the NI LabVIEW RIO Architecture*. Dipetik Januari 17, 2018, dari National Instruments Web site: http://www.ni.com/white-paper/52093/en/

National Instruments. (2016, Mei). *NI myRIO-1900 User Guide and Specifications.* Dipetik Januari 17, 2018, dari National Instruments Web site: http://www.ni.com/pdf/manuals/376047c.pdf

Saha, D., & Mukherjee, A. (2003). Pervasive computing: a paradigm for the 21st century. *Computer, 36(3)*, 25-31.

Shea, S., & Tang, B. A. (2016, November). *What is pervasive computing (ubiquitous computing)?* Dipetik Januari 17, 2018, dari IoT Agenda Website: http://internetofthingsagenda.techtarget.com/definition/pervasive-computing-ubiquitous-computing

Snoeren, A. a. (2000). An End-to-End Approach to Host Mobility. *MobiCom '00 Proceedings of the 6th annual international conference on Mobile Computing and Networking*.

Swastika, V. M. (2015, Juni 17). Diambil kembali dari http://www.kompasiana.com/vanessams/perkembangan-teknologi-di-indonesia_55547634b67e615e14ba545b

technopedia. (2018). *What is State Diagram*. Dipetik April 21, 2018, dari technopedia web site: https://www.techopedia.com/definition/16446/state-diagram

Wigmore, I. (2016, Juli). *What is Internet of Things (IoT)?* Dipetik 1 10, 2018, dari http://whatis.techtarget.com/definition/Internet-of-Things